A Uniform presentation of chocs and p-calculus

Roberto M. Amadio

▶ **To cite this version:**

Roberto M. Amadio. A Uniform presentation of chocs and p-calculus. [Research Report] RR-1726, INRIA. 1992. inria-00076965

HAL Id: inria-00076965
https://hal.inria.fr/inria-00076965

Submitted on 29 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNITÉ DE RECHERCHE
INRIA-LORRAINE

Rapports de Recherche

1 9 9 2

25*ème anniversaire*

N° 1726

Programme 2
Calcul Symbolique, Programmation
et Génie logiciel

A UNIFORM PRESENTATION OF CHOCS AND π-CALCULUS

Roberto M. AMADIO

Juillet 1992

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tél.:(1) 39 63 55 11

RR-1726

A Uniform Presentation of CHOCS and π-calculus

Une Présentation Uniforme de CHOCS et π-calcul

Roberto M. Amadio [1]

CRIN-CNRS, INRIA-Lorraine, Nancy

Abstract

We present a generic calculus of 'mobile' processes intended as language, labelled transition system, and bisimulation. The distinctive feature of this presentation is an explicit treatment of contexts. Calculi having processes (CHOCS) or channels (π-calculus) as transmissible values are obtained as instances of the generic calculus.

Our main technical contributions are:

• A *needed* weakening of the notion of bisimulation for CHOCS and a *new characterization* of π-calculus bisimulation.

• A *sufficient condition* for checking the bisimilarity of CHOCS processes via a standard translation into the π-calculus.

• A *uniform* notion of bisimulation for the π-calculus.

Résumé

Nous présentons un calcul générique des processus 'mobiles' entendu comme langage, système de transition étiqueté et bisimulation. La caractéristique de cette présentation est un traitement explicite des contextes. Les calculs ayants processus (CHOCS) ou canaux (π-calcul) comme valeurs transmissibles sont obtenus comme instances du calcul générique.

Nos contributions techniques principales sont:

• Un affaiblissement *essentiel* de la notion de bisimulation pour CHOCS et une nouvelle caractérisation de la bisimulation du π-calcul.

• Une *condition suffisante* pour contrôler la bisimilarité des processus CHOCS via une traduction standard dans le π-calcul.

• Une notion de bisimulation *uniforme* pour le π-calcul.

[1] Address: URA 262, CRIN-CNRS, B.P. 239, 54506, Vandoeuvre-lès-Nancy, FRANCE. E-mail: amadio@loria.loria.fr.

Introduction

Process calculi are formalisms that describe a collection of entities acting concurrently, and allowed to interact from time to time according to certain communication mechanisms. Describing the *behaviour* of a process means defining its ability to perform certain interactions with the environment. Defining the *equivalence* of two processes means to establish which are the legal observations or experiments that an environment can perform on a process.

Process Calculi such as CCS have proved to be a *useful* tool in the specification of parallel/concurrent programs. A large effort has been dedicated to:

- determine various notions of process equivalence, notably bisimulation and testing.
- define complete theories to reason on process equivalence.
- provide tools for the mechanization of such theories.
- refine the notion of parallelism-as-interleaving to a setting in which it is possible to observe the concurrent execution of several actions.

In spite of their relative success process calculi such as CCS are not free from criticism:

- Their expressive power is limited. The simulation of a Turing machine already asks for a remarkable degree of ingenuity. The representation of modern programming features such as procedures, or abstraction clearly lies outside the practical scope of the calculus.[2]
- The recursion free fragment of process calculi has a very limited dynamic. In order to write interesting programs we are often forced to introduce an unbounded recursion.
- There is no "correct" notion of logical variable and substitution.
- There is no deep understanding of the mechanisms of synchronization or communication. The main problem that comes with this is that there is no "type theory" for process calculi, that is there is no linguistic mechanism that ensures certain remarkable properties of programs such as termination or absence of deadlocks.

Recently introduced extensions of process calculi, such as the π-calculus (Astesiano&Zucca[84], Engberg&Nielsen[86], Milner&al.[89]) or CHOCS (Thomsen[89], but see also Nielsen[88], Boudol[89]) seem to address at least the first two of our criticisms. These calculi will be the object of our study here. They are based on the idea that processes can communicate complex values such as channels and processes, respectively. On one hand this possibility dramatically increases the expressivity of the calculi, for instance reasonable codings of lambda calculi into such process calculi have been proposed. On the other hand the formalization of these calculi presents new technical problems.

[2] In this respect let us point out that the reduction of CCS with value passing to CCS is misleading as it is obtained by the same mechanism that allows to reduce, say, a finitary first order logic to an infinitary propositional logic.

In the first two sections we address the problem of giving a *simple* calculus in which the discipline of static scoping is *adequately* represented. This presentation differs from previous work in the literature in that contexts are explicitly taken into account. This allows, in our opinion, to give a precise and elegant description of the labelled transition system.

In section 3 we develop a general notion of strong (ground) bisimulation for our calculus. We investigate some of its laws and we compare it with the notion of strong (ground) bisimulation presented in the literature. It is shown that our notion gives a well-motivated *weakening* of CHOCS bisimulation (Thomsen[89(a)]) whereas it provides a *new characterization* of π-calculus bisimulation (Milner&al.[89]).

In section 4 we reconsider the relationship between CHOCS and the π-calculus under the light of the newly introduced bisimulation for CHOCS. It is shown that a standard translation from a *variant* of CHOCS into the π-calculus, say $< >$, satisfies the following property:

$$<P> \sim <Q> \Rightarrow P \sim Q,$$

where P, Q are CHOCS processes, $< >$ is the translation mentioned above, and $\sim$ is an appropriate notion of strong bisimulation. The bisimulation for the CHOCS calculus is apparently harder to mechanize than π-calculus bisimulation. Our result opens the way to a sound reduction of CHOCS bisimilarity to π-calculus bisimilarity.[3]

In section 5 we take a closer look at π-calculus bisimulation. Towards a more efficient mechanization we introduce a labelled transition system that explicitly deals with variables. The derived notion of bisimulation enjoys a 'uniformity property'. Roughly this property says that given two equivalent processes P, Q and a transition of P we can find a transition of Q that simulates the one of P, *uniformly*, for all possible substitutions (and symmetrically). We show that *uniform* strong bisimulation *strictly* contains strong bisimulation. The practical fall-out is that in checking bisimulation rather than verifying all 'ground' substitutions we can perform 'symbolic' transitions and delay the actual instances as much as possible. For this reason we expect that the notion of strong uniform bisimulation leads to more efficient mechanizations than that of strong bisimulation.

Contents: Introduction, 1. Terms Formation Rules, 2. Labelled Transition System, 3. Strong Bisimulation, 4. On the Reduction of CHOCS Bisimulation to π-calculus Bisimulation, 5. Bisimulation for π-calculus Revisited. Conclusion. References. Appendix.

[3] *Added in proof:* when this work was finished it was pointed out to us that D. Sangiorgi of the University of Edinburgh has obtained related results in his PhD thesis (which is unfortunately not yet available).

1. Terms Formation Rules

1.1 Scoping Rules

We begin with a perspective on the formalization of the scoping rules from the point of view of an "abstract" machine that executes terms of such process calculi.

• As usual the state of an abstract machine will include a set of pointers for sequence control and data control.[4] One may observe that every time we start to execute two processes in parallel it is natural to split the data list into two branches as the two processes may create local variables in a completely independent way. In this case, as the computation proceeds, data are structured as a tree.

• Let us now consider a process calculus with the following operators:

- Parallel Composition of the processes P and Q: (P | Q).

- Output of a transmissible value t along the channel c and then execution of P: (c ! t. P).

- Input of a transmissible value, say t, along the channel c and then instantiation of such t for the formal parameter x in P and execution of [t/x]P: (c?x. P).

- Creation of a 'vew' channel c local to P: νc.P (we will call this operator restriction).

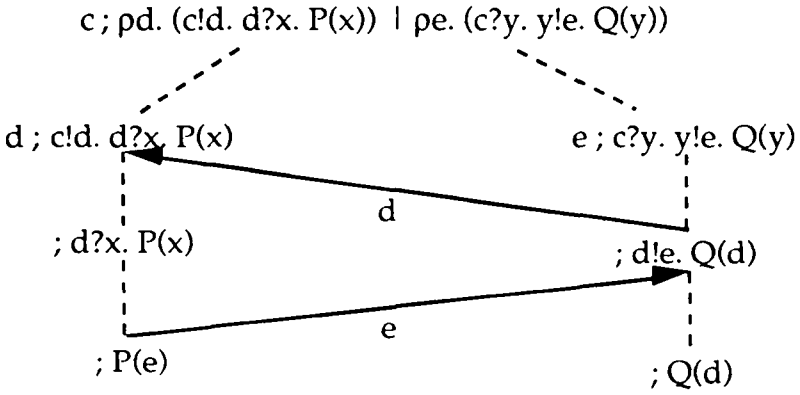
• If the transmissible values do not contain channels that can be bound by the operator "ν" then the data can be really seen as a tree. Attached to the leaves of the tree we may consider the collection of processes that are in parallel execution. The part of the data visible to each of these processes is determined by the path from the corresponding leaf to the root. Two parallel processes share a certain part of data that is given by the intersection of the two corresponding paths from the respective leaves to the root.

• If the transmissible values contain channels then we have to determine which scoping rules are more appropriate. Following a longstanding tradition in statically, strongly typed languages we will concentrate here on *static scoping*. In first approximation this discipline ensures that: (i) the property of being a bound or free variable is invariant under execution of the program, (ii) each bound variable always refers to the same binder during the execution. In order to enforce this discipline in our case, we have to guarantee that upon reception of the value t the process c?x.P(x) becomes P(t) where the body P(_) refers to the same data as c?x.P(x) but t refers to the data relative to the sending process. It appears that a natural way to implement this discipline is to insert *cross references* in the data tree.

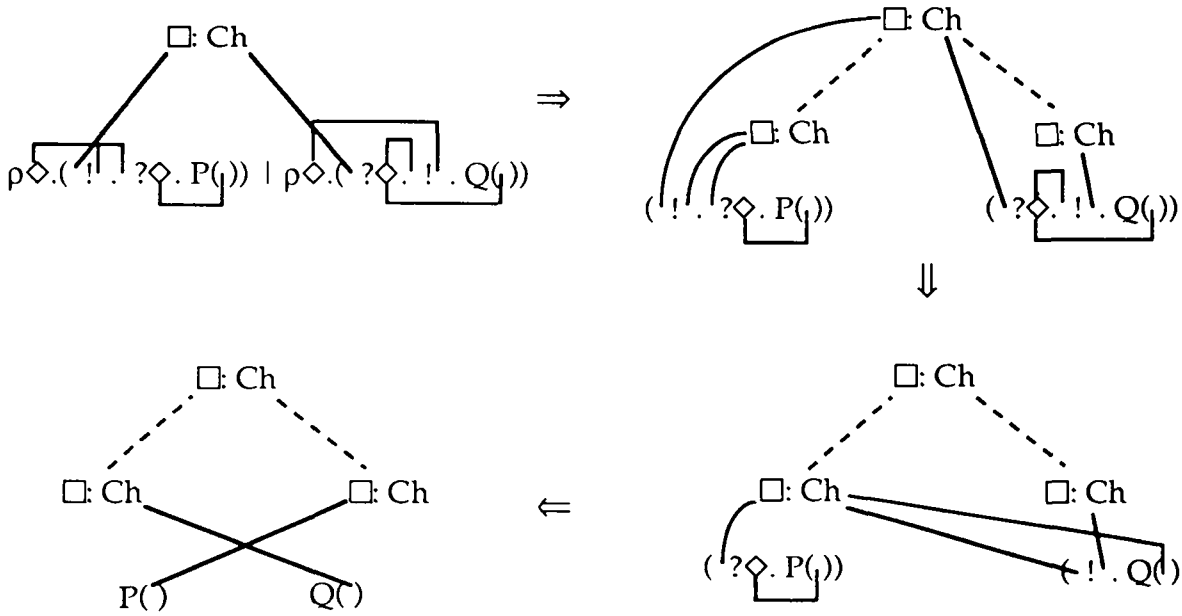
Example: We suppose that channels are transmissible values. On the left of ";" we indicate a channel declaration that is visible to the process on the right of ";" and to its siblings. The dotted lines connect the nodes of the data tree. Full lines labelled

[4] The terms sequence control and data control should be intended as in Pratt[84]. What is called here data list (or tree) is usually called environment. We prefer the former name to avoid confusion with the notion of environment that arises in the literature on reactive systems where one speaks of "the interaction of a process with its environment".

with a channel name indicate exceptions that one has to consider in searching the declaration of a channel.



• It appears that a transmissible value has to be passed with some reference to its data (i.e. a closure). In keeping our view of data as a tree we could represent the data associated to a transmissible value t as a pointer to a node of the tree. Starting from this node and proceeding towards the root we would find all channel declarations relevant for t. The following diagram represents the same computation as above in which variables are "implemented" as pointers to a location. We denote with "□" variables that belong to the data tree and with "◇" variables that are bounded by the operators v and ?. A full line connects a variable occurrence to the corresponding box. The data-tree boxes are arranged along a tree structure whose edges are represented as dotted lines. In the diagram we do not represent *all* the variables that may be free in P and Q.



• This example suggests that there could be a representation of variables as paths in the data tree. Whereas this seems closer to an implementation it would turn the corresponding formalism into something unreadable (we expect something more complex than DeBrujin indices). Another interesting question is which kind of "architecture" could support this kind of communications.

1.2 A Formal Calculus

In this section we introduce the terms of a "generic" process calculus. In such calculus a data tree will be represented as a "context", namely a list of variables (hence we delay the study of calculi closer to the implementation to further investigation). Terms come along with their context and their "kind". This makes syntax more verbose w.r.t. usual presentations of process calculi found in the literature, but it has the advantage of clarifying the structure of the calculus.

Kinds: The expressions of the language are classified in three kinds (K): processes (Pr), channels (Ch), and transmissible values (Tv). Processes have internal activity and exchange transmissible values along channels.

$K ::= Pr \mid Ch \mid Tv$

Language: Channels are names, here represented as variables. Processes may have a complex structure. Transmissible values can be of various kinds, namely values, channels, or processes. Various process calculi will arise by a crisper specification of the nature of transmissible values.

Variables	$v ::= x \mid y \mid \ldots$
Channels	$c ::= v$
Processes	$P ::= ø \mid v \mid (P+P) \mid (P \mid P) \mid (\delta\, P) \mid (vc.P) \mid (c\,!\,t.P) \mid (c\,?\,v.\,P)$
Transm. Values	$t ::= v \mid c \mid P$
Generic Exp.	$exp ::= c \mid P \mid t$

Context: A context Γ is a possibly empty sequence of pairs variable-kind, where all variables are distinct. We explicitly represent a context as: $v_1: K_1, \ldots, v_n: K_n$ $(n \geq 0)$.

Kind Judgments: A kind judgment has the shape: $\Gamma \supset exp: K$, to be read as exp has kind K in the context Γ. A judgment is provable, and we write $\vdash \Gamma \supset exp: K$, if it can be derived in the following formal system, where "$\Rightarrow$" separates the hypotheses from the conclusion.

[asmp]	$v: K \in \Gamma \Rightarrow \Gamma \supset v: K$
[ø]	$\Rightarrow \Gamma \supset ø: Pr$
[+]	$\Gamma \supset P: Pr,\ \Gamma \supset Q: Pr \Rightarrow \Gamma \supset (P+Q): Pr$
[|]	$\Gamma \supset P: Pr,\ \Gamma \supset Q: Pr \Rightarrow \Gamma \supset (P \mid Q): Pr$
[δ]	$\Gamma \supset P: Pr \Rightarrow \Gamma \supset (\delta\, P): Pr$
[v]	$\Gamma, c: Ch \supset P: Pr \Rightarrow \Gamma \supset (vc.P): Pr$
[!]	$\Gamma \supset c: Ch,\ \Gamma \supset t: Tv,\ \Gamma \supset P: Pr \Rightarrow \Gamma \supset (c!t.\,P): Pr$
[?]	$\Gamma \supset c: Ch,\ \Gamma, v: Tv \supset P: Pr \Rightarrow \Gamma \supset (c?v.\,P): Pr$

Proviso *(on α-redenomination and substitution)*

In expressions such as (vc.P) and (d?v.P) the variables c and v, respectively, are bound in P. In the following we identify two terms that differ only by their bound variables. Also we denote with [exp/v]exp' the substitution of exp for v in exp'. This operation has to be carried on using the usual rules of redenomination to avoid that free variables in exp are captured by binders in exp'.

Next we show how we can derive (at the meta-level) rules of context *exchange, weakening, remove* of variables, as well as a *substitution* rule. The proof is a routine induction on the length of the derivations and therefore it is delayed to the appendix.

Conventions: Whenever we write a context Γ we assume that it is well formed. We write $\Gamma \cap \Gamma' = \emptyset$ if the contexts Γ and Γ' share no common variables. If Γ is a context we denote with $\sigma(\Gamma)$ a context resulting from the permutation σ of the pairs in Γ.

1.2.1 Proposition

(1) *Exchange.* If $\vdash\Gamma \supset$ exp: K then $\vdash\sigma(\Gamma) \supset$ exp: K.
(2) *Weakening.* If $\vdash\Gamma \supset$ exp: K and $\Gamma \cap \Gamma' = \emptyset$ then $\vdash\Gamma,\Gamma' \supset$ exp: K.
(3) *Removing Variables.* If $\vdash\Gamma,v{:}K \supset$ exp: K and $v \notin$ exp then $\vdash\Gamma \supset$ exp: K.
(4) *Substitution.* If $\vdash\Gamma,\Gamma',v{:}K_1 \supset$ exp: K, $\vdash\Gamma,\Gamma_1 \supset$ exp$_1$: K_1, Γ,Γ',Γ_1 well-formed then $\vdash\Gamma,\Gamma',\Gamma_1 \supset$ [exp$_1$/v] exp: K.

2. Labelled Transition System

Type of Actions: Processes are described by their ability to perform actions. In first approximation such actions can be of three types: (i) an internal action (τ), (ii) an input communication along a given channel (c?), (iii) an output communication of a transmissible value along a given channel (c!t).

Static Scoping: Particular care has to be put in the transmission of non-elementary transmissible values, such as channels or processes, which may depend on the context. As discussed in the first section we wish to define the semantics according to the rules of *static scoping*. In order to represent these rules one may think to transmit a context along with the value t. However it turns out that the context associated with t can be recovered by a suitable manipulation of the rules, this is the reason why in the output action only t appears.

Actions: Hence we consider the following actions

$\alpha ::= \tau \mid c? \mid c!t$

2.1 Labelled Transition System: We define a formal system to derive judgments of the shape: $\Gamma \supset P \mapsto\alpha\ \Gamma' \supset P'$

that should be read as "from the process P in the context Γ we make an action α and we generate a process P' in the context Γ' ". The application of the rules is in general non-deterministic but it is guided by the structure of the process P. This kind of systems are often called labelled transition systems (lts).

Conventions: We write $c \in t$ if the channel c occurs free in the transmissible value t. We also write $c \notin \alpha$ if c is not free in the action α, i.e. $\alpha \equiv d? \Rightarrow d \neq c$ and $\alpha \equiv d!t \Rightarrow d \neq c$ and $c \notin t$. If $\Gamma \equiv c_1: Ch, ..., c_m: Ch$ $(m \geq 0)$ then $\nu\Gamma \equiv \nu c_1 ... \nu c_m$. No rules are associated to a variable or to the terminated process ø. We omit to write the rules symmetric to: (+.left), (|.left), (|.com.!?).

(!) $$\frac{}{\Gamma \supset (c!t.\ P) \mapsto c!t\ \ \Gamma \supset P}$$

(?) $$\frac{}{\Gamma \supset (c?v.\ P) \mapsto c?\ \Gamma, v:Tv \supset P}$$

(+.left) $$\frac{\Gamma \supset P \mapsto\alpha\ \Gamma' \supset P'}{\Gamma \supset (P+Q) \mapsto\alpha\ \Gamma' \supset P'}$$

(|.left) $$\frac{\Gamma \supset P \mapsto\alpha\ \Gamma' \supset P'}{\Gamma \supset (P \mid Q) \mapsto\alpha\ \Gamma' \supset (P' \mid Q)}$$

(δ) $$\frac{\Gamma \supset P \mid P \mapsto\alpha\ \Gamma' \supset P'}{\Gamma \supset \delta P \mapsto\alpha\ \Gamma' \supset P' \mid \delta P}$$ [5]

(ν) $$\frac{\Gamma, c: Ch \supset P \mapsto\alpha\ \Gamma, c: Ch, \Gamma' \supset P' \quad c \notin \alpha}{\Gamma \supset (\nu c.P) \mapsto\alpha\ \Gamma, \Gamma' \supset (\nu c.P')}$$

(ν.open) $$\frac{\Gamma, c: Ch \supset P \mapsto d!t\ \ \Gamma, c: Ch, \Gamma'' \supset P' \quad c \in t,\ d \neq c}{\Gamma \supset (\nu c.P) \mapsto d!t\ \ \Gamma, c: Ch, \Gamma'' \supset P'}$$

(|.com!?) $$\frac{\Gamma \supset P \mapsto d!t\ \ \Gamma, \Gamma' \supset P' \quad \Gamma \supset Q \mapsto d?\ \Gamma, v:Tv \supset Q'}{\Gamma \supset (P \mid Q) \mapsto\tau\ \Gamma \supset \nu\Gamma'.(P' \mid [t/v]Q')}$$

[5] We choose this formulation of the rule because it induces the nice properties presented in section 3 as 'δ-laws' while making clear that the collection of transitions that can be performed by any given process are finite and computable.

Notes

(1) The assumptions on the structure of the contexts in the rules (ν.open) and (|.com.!?) may appear unduly restrictive. For example in (|.com!?) one may think to write: $\Gamma \supset Q \mapsto_{d?} \Gamma, v{:}Tv, \Gamma' \supset Q'$. However it would then become a meta-theorem that $\Gamma' = \emptyset$. For the sake of simplicity we have directly introduced here the "optimized" system.

(2) The crucial rules are (ν.open) and (|.com!?). The rule (ν.open) removes the restriction "νc" from the process νc.P whenever such process sends a transmissible value t containing c. This is because external agents by manipulating t *may* then be able to access c. The manipulation of the contexts in order to achieve the phenomena of exporting a local channel (also called scope extrusion) is then performed by the rule (|.com!?). In this case Γ'' contains exactly the channel variables local to P on which the transmitted value t depends. By the rule (ν.open) these channel variables have been "opened" in P', we therefore just have to "close" them again while including [t/v]Q' in their scope.

The following proposition relates actions and contexts, and it states that well typed processes are closed under reduction in the lts presented above. The proof is a routine induction on the structure of a derivation in the lts, and is therefore delayed to the appendix.

2.2 Proposition

(1) If $\vdash \Gamma \supset P \mapsto_\alpha \Gamma' \supset P'$ then $\Gamma \subseteq \Gamma'$, and moreover
  if $\alpha \equiv \tau$ then $\Gamma \equiv \Gamma'$.
  if $\alpha \equiv d?$ then $\Gamma, x{:}Tv \equiv \Gamma'$ for some x.
  if $\alpha \equiv d!t$ then $\Gamma, c_1{:}\,Ch, \ldots, c_m{:}\,Ch \equiv \Gamma'$, for some $m \geq 0$, $c_1, \ldots, c_m$.

(2) If $\vdash \Gamma \supset P{:}\,Pr$ and $\vdash \Gamma \supset P \mapsto_\alpha \Gamma, \Gamma' \supset P'$ then
  (a) $\vdash \Gamma, \Gamma' \supset P'{:}\,Pr$, and
  (b) if $\alpha \equiv d!t$ then $\vdash \Gamma, \Gamma' \supset t{:}\,Tv$.

3. Strong Bisimulation

In this section we first develop a notion of strong *ground* bisimulation (sgb) for our generic calculus and we compare it with the analogous notions of sgb already proposed for CHOCS and π-calculus. The adjective 'ground' here refers to the fact that all assumptions in the context are considered as constants.

Next we introduce a notation to represent assumptions on variables and we state that two processes are strongly bisimilar if they are strongly *ground* bisimilar for all possible ground instances of the assumptions on variables.

3.1 Strong Ground Bisimulation

Conventions

• We denote with $Pr(\Gamma) \triangleq \{P \mid \vdash \Gamma \supset P: Pr\}$ the collection of processes to which we can assign a kind in the context Γ. If Cont is the set of well formed contexts then we denote with S, T,... a family of binary relations indexed over Cont, such that:

$S: Cont \to \cup_{\Gamma \in Cont} Pr(\Gamma)^2$, $S(\Gamma) \subseteq Pr(\Gamma)^2$, for any Γ.

If $(P, Q) \in S(\Gamma)$ then we write more suggestively $\Gamma \supset PSQ$. Note that whenever we write $\Gamma \supset PSQ$ it is expected $\vdash \Gamma \supset P: Pr$ and $\vdash \Gamma \supset Q: Pr$.

• In some examples we will write processes without caring for their context. In these cases a context can be easily found so that the examples make sense in our formal setting. We also write d? and d! whenever the input/output values do not matter. Formally one can imagine: $d?.P \equiv d?x.P$, where $x \notin P$, and $d!.P \equiv d!t.P$, where t is some globally visible transmissible value.

3.1.1 Definition *(strong ground bisimulation)*

A relation S is a sgb if $\Gamma \supset P\ S\ Q$ and for any $\vdash \Gamma \supset P \mapsto_\alpha \Gamma' \supset P'$ then:

(τ) $\alpha \equiv \tau$ implies there is $\vdash \Gamma \supset Q \mapsto_\tau \Gamma \supset Q'$, $\Gamma \supset P'SQ'$.

(?) $\alpha \equiv d?$, $\Gamma' \equiv \Gamma, x:Tv$, $\vdash \Gamma, \Gamma'' \supset t:Tv$ implies there is
$\vdash \Gamma \supset Q \mapsto_{d?} \Gamma, x:Tv \supset Q'$, $\Gamma, \Gamma'' \supset [t/x]P'\ S\ [t/x]Q'$,

(!) $\alpha \equiv d!t$, $\Gamma' \equiv \Gamma, \Gamma_1$, $\vdash \Gamma, \Gamma'', x:Tv \supset R:Pr$, $\Gamma'' \cap \Gamma_1 = \emptyset$ implies there is
$\vdash \Gamma \supset Q \mapsto_{d!s} \Gamma, \Gamma_2 \supset Q'$, $\Gamma'' \cap \Gamma_2 = \emptyset$, $\Gamma, \Gamma'' \supset \nu\Gamma_1.(P' \mid [t/x]R)\ S\ \nu\Gamma_2.(Q' \mid [s/x]R)$.

and *symmetrically*.

3.1.2 Proposition *(basic properties of sgb)*

(1) Sgbs are closed under arbitrary (indexed) unions.

(2) We write $\Gamma \supset P \sim Q$ iff $\Gamma \supset PSQ$, for some sgb S. The following equivalences hold:

Monoidal Laws:

(+.id)	$\Gamma \supset (P+\emptyset) \sim P$	(|.id)	$\Gamma \supset (P \mid \emptyset) \sim P$
(+.assoc)	$\Gamma \supset (P+P')+P'' \sim P+(P'+P'')$	(|.assoc)	$\Gamma \supset (P \mid P') \mid P'' \sim P \mid (P' \mid P'')$
(+.com)	$\Gamma \supset (P+P') \sim (P'+P)$	(|. com)	$\Gamma \supset (P \mid P') \sim (P' \mid P)$
(+.idem)	$\Gamma \supset (P+P) \sim P$		

Restriction Commutations:

(ν.free)	$\Gamma \supset \nu c.P \sim P$, if $c \notin P$	(ν.+)	$\Gamma \supset \nu c.P+\nu c.P' \sim \nu c.(P+P')$.
(ν.!-ø)	$\Gamma \supset \nu c.c!t.P \sim \emptyset$	(ν.|)	$\Gamma \supset (\nu c.P) \mid P' \sim \nu c.(P \mid P')$, if $c \notin P'$
(ν.?-ø)	$\Gamma \supset \nu c.c?x.P \sim \emptyset$	(ν.!)	$\Gamma \supset (d!t.\ \nu c.P) \sim \nu c.(d!t.\ P)$, if $c \notin t$
(ν.ν)	$\Gamma \supset \nu d.\nu c.P \sim \nu c.\nu d.P$	(ν.?)	$\Gamma \supset (d?v.\ \nu c.P) \sim \nu c.(d?v.\ P)$

Duplication Laws:

(δ_1)	$\Gamma \supset \delta P \sim P \mid \delta P$	(δ_2)	$\Gamma \supset \delta P \sim \delta P \mid \delta P$
($\delta\delta$)	$\Gamma \supset \delta P \sim \delta(\delta P)$	(δ.|)	$\Gamma \supset \delta(P \mid Q) \sim \delta P \mid \delta Q$

Congruence Rules:

(exch) $\Gamma \supset P \sim Q \Rightarrow \sigma\Gamma \supset P \sim Q$

(weak) $\Gamma \supset P \sim Q \Rightarrow \Gamma, \Gamma' \supset P \sim Q$

(+) $\Gamma \supset P_1 \sim Q_1,\ \Gamma \supset P_2 \sim Q_2 \Rightarrow \Gamma \supset P_1+P_2 \sim Q_1+Q_2$

(|) $\Gamma \supset P_1 \sim Q_1,\ \Gamma \supset P_2 \sim Q_2 \Rightarrow \Gamma \supset P_1 \mid P_2 \sim Q_1 \mid Q_2$

(ν) $\Gamma, c{:}Ch \supset P \sim Q \Rightarrow \Gamma \supset \nu c.P \sim \nu c.Q$

(δ) $\Gamma \supset P \sim Q \Rightarrow \Gamma \supset (\delta P) \sim (\delta Q)$

(!) $\Gamma \supset P \sim Q \Rightarrow \Gamma \supset c!t.P \sim c!t.Q$

(?) for any $\vdash \Gamma, \Gamma'' \supset t{:}Tv,\ \Gamma, \Gamma'' \supset [t/x]P \sim [t/x]Q \Rightarrow \Gamma \supset c?x.P \sim c?x.Q$

Proof

(1) If $\{S_i\}_{i \in I}$ is a family of sgbs then the indexed union is the sgb S defined as $\Gamma \supset P\ S\ Q$ if $\Gamma \supset P\ S_i\ Q$ for some $i \in I$.

(2) The monoidal laws are typical of the CCS family of process calculi, they are proven by an easy analysis of the actions that can be performed on the two sides of the equivalence. The restriction commutations can be verified in a similar fashion. The duplication laws are verified in the order (δ_1), $(\delta\delta)$, (δ_2), $(\delta.\mid)$ by some case analysis of the transitions. The basic reason why these equivalences are satisfied is that even if we have an arbitrary number of copies of P the actions that can be actually performed are just those arising either from *one* copy of P or from the communication of *two* copies of P. The congruence for (|) deserves some attention, one proves first the one-side congruence: $\Gamma \supset P \sim Q \Rightarrow \Gamma \supset P \mid R \sim Q \mid R$. □

Notes

(1) Concerning some *commutations between restriction and duplication* the following equivalences hold:

(ν.δ) $\Gamma \supset \nu d.\delta(d?x.\ P) \sim \emptyset \qquad \Gamma \supset \nu d.\delta(d!t.\ P) \sim \emptyset.$

They are both derived from the following rule:

(δ) $\Gamma \supset \nu c.P \sim \emptyset \Rightarrow \Gamma \supset \nu c.(\delta P) \sim \emptyset.$ □

(2) There is also a rather complex to write *'expansion law'*. In the following we will just use some trivial instances of it. The expansion law also has a particular place as it depends on the reduction of concurrent actions to their interleaving which we have implicitly formulated in the lts. On the other hand axioms and rules stated above should be stable under reasonable modifications of the lts. □

3.2 Motivating Strong Ground Bisimulation

We propose some examples which explain the origin of clauses (?) and (!) in sgb definition.

• Clause (?) should be interpreted as saying that if P can perform an input action along the channel d then also Q can, and moreover whatever value, say t, P and Q receive, [t/x]P' and [t/x]Q' continue to bisimulate. One may wonder why it is not enough to check the equivalence of P'(x) and Q'(x), rather than trying all the inputs.[6] One problem is that variables are not really treated as variables in the

(Com.!?) rule of the lts. For instance, suppose that transmissible values include channels. Then we expect the equivalence:

$$x? \mid d! \sim x?.\, d! + d!.\, x?$$

by an application of the expansion law. But observe that if x is an input-variable then x may happen to be d, and in this case a communication is possible in x? | d!. For this reason the equivalence:

$$c?x.\, (x? \mid d!) \sim c?x.\, (x?.\, d! + d!.\, x?)$$

turns out to be *incorrect*. In the current formulation of the lts, clause (?) is a simple way to get a sound equivalence, but we believe that alternative formulations should be studied (see also section 5).

• Clause (!) express the fact that when a process sends a transmissible value to the environment it may also export part of the information on its local channels (scope extrusion). To justify the complexity of the clause let us consider two extreme formulations of the (!) clause:

($!_1$) $\alpha \equiv d!t$, $\Gamma' \equiv \Gamma,\Gamma_1$ implies there is $\vdash \Gamma \supset Q \mapsto^{d!s} \Gamma,\Gamma_2 \supset Q'$, $\Gamma \supset \nu\Gamma_1.P'\ S\ \nu\Gamma_2.Q'$ (plus conditions on the equivalence of s and t).

In this case we ignore the phenomena of scope extrusion. Consider:

$$P \equiv \nu c.a!c.c!.P', \text{ and } Q \equiv \nu c.a!c.\emptyset$$

Then from $\nu c.c!.P' \sim \emptyset$ we could conclude $P \sim Q$, that is incorrect since the environment having received the channel c can perform a communication along it and enable the execution of an arbitrary process P'. Next consider the clause:

($!_2$) $\alpha \equiv d!t$, $\Gamma' \equiv \Gamma, \Gamma_1$, implies there is $\vdash \Gamma \supset Q \mapsto^{d!s} \Gamma, \Gamma_2 \supset Q'$, $\Gamma,\Gamma_1, \Gamma_2 \supset P'\ S\ Q'$ (modulo a "suitable" identification of the local channels Γ_1 and Γ_2, plus conditions on the equivalence of t and s).

In this case we seem to admit the existence of a very powerful environment that having received a value t can *directly* manipulate all its local channels. We do not consider this as a natural hypothesis. In the next example, that we learned from E. Moggi, we show that this hypothesis can make a difference in the definition of process equivalence. Suppose processes are transmissible values and consider:

$$P \equiv \nu c.\nu d.a!(d?.c?.\emptyset).c!.P', \text{ and } Q \equiv a!\emptyset.\emptyset.$$

A rule in the style of ($!_2$) would distinguish P and Q by observing that c!.P is not equivalent to ø. But suppose the environment is just a process in our calculus, say a?x.R, and observe that we expect:

$$\nu c.\nu d.(c!.P' \mid [d?.c?.\emptyset/x]R) \sim [\emptyset/x]R \qquad (1)$$

The intuition is that since the d? action is restricted we never arrive to make a communication on the c channel. The reader may try to define a bisimulation showing (1). We will give an indirect formal proof of a variant of (1) in section 4. We conclude that clauses in the style of ($!_2$), although attractive for their simplicity,

[6] This would be reminiscent of the ξ-rule in λ-calculus: $\lambda x.M = \lambda x.N$ if $M=N$.

may fail to give a *complete* theory of process equivalence. □

3.3 Comparison with CHOCS

We refer to Thomsen[89(a)] where a version of static CHOCS is introduced. We represent CHOCS programs by supposing that the kinds of transmissible values and processes coincide: Tv≡Pr. Then we can state Thomsen's notion of sgb as follows:

- clauses (τ) and (?) are the same.
- (!$_{CH}$) $\alpha \equiv d!P_1$, $\Gamma' \equiv \Gamma,\Gamma_1$ implies there is
  $\vdash \Gamma \supset Q \mapsto^{d!Q_1} \Gamma,\Gamma_1 \supset Q'$, $\Gamma,\Gamma_1 \supset P' \, S \, Q'$, $\Gamma,\Gamma_1 \supset P_1 \, S \, Q_1$.

We denote with $\sim_{CH}$ the largest of these sgbs. We now show that the equivalence $\sim_{CH}$ is contained in $\sim$.

3.3.1 Proposition *(weakening CHOCS bisimulation)*
If $\Gamma \supset P \sim_{CH} Q$ then $\Gamma \supset P \sim Q$.

Proof

We show that $\sim_{CH}$ is a sgb. Suppose $\Gamma \supset P \sim_{CH} Q$ and $\vdash \Gamma \supset P \mapsto^{\alpha} \Gamma' \supset P'$. We now analyse the possible actions:

(τ) $\alpha \equiv \tau$. Then by definition of CH-sgb $\vdash \Gamma \supset Q \mapsto^{\tau} \Gamma \supset Q'$, $\Gamma \supset P' \sim_{CH} Q'$.
(?) same argument as for (τ).

(!) $\alpha \equiv d!P_1$, $\Gamma' \equiv \Gamma, \Gamma_1$, $\vdash \Gamma,\Gamma'',x{:}Pr \supset R{:}Pr$, $\Gamma'' \cap \Gamma_1 = \emptyset$.
By definition of CH-sgb we have:
$\vdash \Gamma \supset Q \mapsto^{d!Q_1} \Gamma,\Gamma_1 \supset Q'$, $\Gamma,\Gamma_1 \supset P' \sim_{CH} Q'$, $\Gamma,\Gamma_1 \supset P_1 \sim_{CH} Q_1$. Next we observe:
- $\Gamma,\Gamma_1 \supset P_1 \sim_{CH} Q_1$, $\vdash \Gamma,\Gamma'',x{:}Pr \supset R{:}Pr$ implies $\Gamma,\Gamma_1,\Gamma'' \supset [P_1/x]R \sim_{CH} [Q_1/x]R$.
- $\Gamma,\Gamma_1,\Gamma'' \supset [P_1/x]R \sim_{CH} [Q_1/x]R$, $\Gamma,\Gamma_1 \supset P' \sim_{CH} Q'$ implies
  $\Gamma,\Gamma_1,\Gamma'' \supset (P' \mid [P_1/x]R) \sim_{CH} (Q' \mid [Q_1/x]R)$ implies
  $\Gamma,\Gamma'' \supset \nu\Gamma_1.(P' \mid [P_1/x]R) \sim_{CH} \nu\Gamma_1.(Q' \mid [Q_1/x]R)$. □

3.4 Comparison with the π-calculus

We refer to Milner&al.[89] where a version of (static) π-calculus is introduced. We represent π-calculus programs by supposing that the kinds of transmissible values and channels coincide, Tv≡Ch, and moreover that variables in the context are not of the process kind. We can then state their notion of sgb as follows:

- clauses (τ) and (?) are the same.
- (!$_{\pi}$) $\alpha \equiv d!c$, $\Gamma' \equiv \Gamma,\Gamma_1$ implies there is
  $\vdash \Gamma \supset Q \mapsto^{d!c} \Gamma,\Gamma_1 \supset Q'$, $\Gamma,\Gamma_1 \supset P' \, S \, Q'$.

We denote with $\sim_{\pi}$ the largest of these bisimulations. We now show that the equivalence $\sim_{\pi}$ is the same as the one obtained as an instance of $\sim$.

3.4.1 Proposition
If $\Gamma \supset P \sim_{\pi} Q$ then $\Gamma \supset P \sim Q$.

Proof

Like for CHOCS we verify that $\sim_\pi$ is a sgb. Suppose $\Gamma \supset P \sim_\pi Q$ and $\vdash \Gamma \supset P \mapsto_\alpha \Gamma' \supset P'$. The interesting case is (!):

(!) $\alpha \equiv d!c$, $\Gamma' \equiv \Gamma, \Gamma_1$, $\vdash \Gamma,\Gamma'',x{:}Ch \supset R{:}Pr$, $\Gamma'' \cap \Gamma_1 = \emptyset$. By definition of π-sgb we have: $\vdash \Gamma \supset Q \mapsto_{d!c} \Gamma,\Gamma_1 \supset Q'$, $\Gamma,\Gamma_1 \supset P' \sim_\pi Q'$. Next we observe:

$\Gamma,\Gamma_1 \supset P' \sim_\pi Q' \vdash \Gamma,\Gamma'',x{:}Ch \supset R{:}Pr$ implies

$\Gamma,\Gamma_1,\Gamma'' \supset (P' \mid [c/x]R) \sim_\pi (Q' \mid [c/x]R)$ implies

$\Gamma,\Gamma'' \supset \nu\Gamma_1.(P' \mid [c/x]R) \sim_\pi \nu\Gamma_1.(Q' \mid [c/x]R)$. □

3.4.2 Lemma

If $\Gamma \supset P \sim Q$ and $\vdash \Gamma \supset P \mapsto_{d!c} \Gamma' \supset P'$ then either

(i) $\Gamma' \equiv \Gamma$, $\vdash \Gamma \supset Q \mapsto_{d!c} \Gamma \supset Q'$, $\Gamma \supset P' \sim Q'$, or

(ii) $\Gamma' \equiv \Gamma,c{:}Ch$, $\vdash \Gamma \supset Q \mapsto_{d!c} \Gamma' \supset Q'$, $\Gamma,\Gamma'' \supset \nu c.(P' \mid [c/x]R) \sim \nu c.(Q' \mid [c/x]R)$, any $\vdash \Gamma,\Gamma'',x{:}Ch \supset R{:}Pr$.

Proof

(i) Suppose $\Gamma' \equiv \Gamma$, $\vdash \Gamma,\Gamma'',x{:}Ch \supset R{:}Pr$. There are two cases to consider:

- Q also emits a global channel, say:

$\vdash \Gamma \supset Q \mapsto_{d!c'} \Gamma \supset Q'$ and $\Gamma,\Gamma'' \supset (P' \mid [c/x]R) \sim (Q' \mid [c'/x]R)$.

Take $R \equiv x!a!$, where $a \in \Gamma''$. Then necessarily $c=c'$, o.w. $P' \mid c!a!$ and $Q' \mid c'!a!$ fail to bisimulate (make a c! action and then an a! action). Then consider $R \equiv \emptyset$, to prove: $\Gamma \supset P' \sim Q'$.

- Q emits a local channel. This case is impossible, say:

$\vdash \Gamma \supset Q \mapsto_{d!c'} \Gamma, c' \supset Q'$ and $\Gamma,\Gamma'' \supset (P' \mid [c/x]R) \sim \nu c'.(Q' \mid [c'/x]R)$.

Take again $R \equiv x!a!$, where $a \in \Gamma''$. Then $P' \mid c!a!$ and $\nu c'.(Q' \mid c'!a!)$ fail to bisimulate as above.[7]

(ii) Suppose $\Gamma' \equiv \Gamma,c{:}Ch$, $\vdash \Gamma,\Gamma'',x{:}Ch \supset R{:}Pr$. Up to symmetries we have to consider:

- Q emits a local channel. Then, up to redenomination, we can identify the two channels and conclude as in (ii). □

3.4.3 Lemma

If $\vdash \Gamma \supset P{:}Pr$, $\vdash \Gamma \supset Q{:}Pr$, $\Gamma,\Gamma'' \supset \nu c.(P \mid [c/x]R) \sim \nu c.(Q \mid [c/x]R)$ any $\vdash \Gamma,\Gamma'',x{:}Ch \supset R{:}Pr$, then $\Gamma,c{:}Ch \supset P \sim Q$.

Proof

We delay this long and technical proof to the appendix. The idea is to build R in such a way that P and Q 'do not realize' that their c channel is restricted. □

3.4.4 Theorem *(characterization of π-calculus sgb)*

If $\Gamma \supset P \sim Q$ then $\Gamma \supset P \sim_\pi Q$.

---

[7] This kind of argument does not seem to go through when considering weak bisimulation. Hence in this case it remains to be seen if the two notions of sgb coincide.

Proof

We show that ~ is a π-sgb. Clearly the interesting cases arises when we make an output action.

- If $\vdash \Gamma \supset P \mapsto_{d!c} \Gamma \supset P'$ then apply lemma 3.4.2, case (i).
- If $\vdash \Gamma \supset P \mapsto_{d!c} \Gamma, c{:}Ch \supset P'$ then apply lemma 3.4.2, case (ii) to have:

  $\Gamma, \Gamma'' \supset \nu c.(P' \mid [c/x]R) \sim \nu c.(Q' \mid [c'/x]R)$, any R

and then conclude by lemma 3.4.3:

$\Gamma, c{:}Ch \supset P' \sim Q'$. □

3.5 Strong Bisimulation

As we have emphasized at the beginning of this section the notion of sgb treats all assumptions as constants. We now wish to extend our notions of bisimulation to processes containing free variables.

Convention: We underline the assumptions in a context that have to be considered as referring to variables. E.g. $\underline{x{:}Ch}$, y:Pr, $\underline{z{:}Ch}$. Henceforth we denote with Γ, Γ',... contexts that do not contain underlined assumptions, and with Δ, Δ',... contexts that *may* contain underlined assumptions. If Γ is a context $\underline{\Gamma}$ denotes the same context where we underline all assumptions.

3.5.1 Definition *(strong bisimulation)*

$\Gamma, \underline{x_1{:}K_1}, \ldots, \underline{x_n{:}K_n} \supset P \sim Q$ if for any $\Gamma, \Gamma' \supset exp_i{:}K_i$ (i=1,..,n),
$\Gamma, \Gamma' \supset [exp_1/x_1, \ldots, exp_n/x_n]P \sim [exp_1/x_1, \ldots, exp_n/x_n]Q$.[8]

Notes *(on strong bisimulation)*

(1) $\Delta, \underline{x{:}K} \supset P \sim Q$ implies $\Delta, x{:}K \supset P \sim Q$ (but not vice versa).

(2) All the laws stated for sgb are valid if we substitute Γ with Δ.

(3) The congruence rule for (?) can now be stated as follows:

$\Delta, \underline{x{:}Tv} \supset P \sim Q \Rightarrow \Delta \supset c?x.P \sim c?x.Q$.

4. On the Reduction of CHOCS Bisimulation to π-calculus Bisimulation

In Thomsen[89(a)] a natural translation from CHOCS into the π-calculus is presented. The basic idea of the translation is that we simulate the delivery of a process by sending the name of a special channel that works like an activator for the process to be sent.

How does this translation behave w.r.t. the notion of sgb ? In the first place we introduce, following Thomsen[89(a)], a *variant* of CHOCS where carrying on a substitution "costs" an extra τ-action. We denote this system with CH_τ. Thomsen[89(a)] observes that the translation preserves his notion of CH_τ-sgb (which is contained in ours). However he also notes that the translation *is not* injective, up-to-bisimulation. Our contribution here is to prove that the translation *is*

[8] The notion of strong bisimulation (like the one of sgb) is invariant under permutations of the assumptions.

injective up to our notion of bisimulation for CHOCS. This fact provides a *sufficient* condition to check CH_τ-sgb, the condition seems to be quite good, at least to the point that we do not know if it is also necessary.

On the other hand this result is not always helpful in reasoning about the original CHOCS calculus (without extra τ-actions). As a matter of fact we will observe in simple examples that the introduction of extra τ-actions can completely jeopardize the notion of equivalence.

4.1 Translation from CHOCS to π-calculus

We define a translation $< >$ by induction on the structure of contexts, terms, and judgments as follows:

$<\emptyset> = \emptyset$, $<x: Pr, \Gamma> = x: Ch, <\Gamma>$, $<x: Ch, \Gamma> = x: Ch, <\Gamma>$

$<\varnothing> = \varnothing$, $<P+P'> = <P>+<P'>$, $<P \mid P'> = <P> \mid <P'>$,

$<\delta P> = \delta <P>$ $<\nu c.P> = \nu c.<P>$, $<c?x.P> = c?x.<P>$

$<c!P'.P> = \nu d.(c!d.\ (<P> \mid \delta(d?.<P'>))$ for $d \notin P, P'$, $<x> = x!$

$<\Gamma \supset P:Pr> = <\Gamma> \supset <P>:Pr$

4.1.1 Proposition

If $\vdash \Gamma \supset P:Pr$ then $\vdash <\Gamma \supset P:Pr>$.

Proof

This is a simple induction on the length of the judgment derivation.

[asmp] If $x: Pr \in \Gamma \Rightarrow \Gamma \supset x: Pr$, then $\vdash <\Gamma> \supset x!:Pr$.

[ø], [+], [|], [δ], [ν]: immediate.

[!] Suppose: $\Gamma \supset c: Ch$, $\Gamma \supset P': Pr$, $\Gamma \supset P: Pr \Rightarrow \Gamma \supset (c!P'.\ P): Pr$. Then build a derivation of $<\Gamma> \supset \nu d.(c!d.\ (<P> \mid \delta(d?.<P'>)): Pr$ (for $d \notin P, P'$) from the derivations of $<\Gamma> \supset c:Ch$, $<\Gamma> \supset <P'>:Pr$, $<\Gamma> \supset <P>: Pr$.

[?] If $\Gamma \supset c: Ch$, $\Gamma, v: Pr \supset P: Pr \Rightarrow \Gamma \supset (c?v.\ P): Pr$, then
$<\Gamma> \supset c: Ch$, $<\Gamma>, v: Ch \supset P: Pr \Rightarrow \Gamma \supset (c?v.\ P): Pr$. □

4.2 τ-Substitution

If we compare the computation of a CHOCS process P and of its translation $<P>$ we observe that in the π-calculus the activation of a process previously sent costs an extra τ action. In order to try to keep P and $<P>$ in lockstep it is therefore useful to introduce the following variant of CHOCS.

4.2.1 Definition *(CHOCSτ-calculus)*

- Syntax. Add the clause: $P ::= \tau.P$.
- Kind. Add the rule: $\Gamma \supset P: Pr \Rightarrow \Gamma \supset \tau.P: Pr$.
- τ-substitution: the substitution $[Q/x]_\tau P$ is defined exactly as $[Q/x]P$ except when $P \equiv x$; in this case $[Q/x]_\tau x \equiv \tau.Q$.

• Lts. Add the axiom: $\Gamma \supset \tau.P \mapsto_\tau \Gamma \supset P$.
Moreover in the (| .com!?) rule use the τ-substitution.

• Strong (Ground) Bisimulation is now defined as in section 3 but using τ-substitution. To emphasize that we are dealing with this variant of CHOCS we write the related bisimulation as "$\sim_\tau$".

The following definition of {/} aims at simulating the notion of process substitution in the calculus:

$$\{Q/x\}P \equiv \nu d.([d!/x]P \mid \delta(d?.Q)) \qquad (\text{for } d \notin P, Q).$$

The following lemmas show that storing the term Q and activating it when needed via an extra communication as done in {Q/x}P is equivalent to make an extra τ action for each copy.

4.2.2 Lemma *(simulating τ-substitution)*
If $\vdash \Gamma,\Gamma',x{:}Pr \supset P{:}\ Pr$, $\vdash \Gamma,\Gamma_1 \supset Q{:}\ Pr$, Γ,Γ',Γ_1 well-formed then
$\underline{\Gamma,\Gamma',\Gamma_1} \supset \{Q/x\}P \sim_\tau [Q/x]_\tau P$.

Proof

It is easy to verify that the kinds can be correctly assigned. Next we proceed by induction on the length of the kind judgment for P. We will omit to write the part of the context that is not relevant for the manipulations carried on.

[asmp] If $P \equiv y \neq x$ then $\{Q/x\}y \equiv \nu d.(y \mid \delta(d?.Q)) \sim_\tau y \equiv [Q/x]_\tau y$.
If $P \equiv y \equiv x$ then $\{Q/x\}x \equiv \nu d.(d! \mid \delta(d?.Q)) \sim_\tau \tau.Q \equiv [Q/x]_\tau x$.

[ø] $\{Q/x\}ø \equiv \nu d.(ø \mid \delta(d?.Q)) \sim_\tau ø \equiv [Q/x]_\tau ø$.

[+] $\{Q/x\}P_1+P_2 \equiv \nu d.([d!/x]P_1+[d!/x]P_2 \mid \delta(d?.Q))$.
$[Q/x]_\tau P_1+P_2 \equiv [Q/x]_\tau P_1+[Q/x]_\tau P_2 \sim_\tau \{Q/x\}P_1+\{Q/x\}P_2$ (by ind. hyp. and cong.)

We prove the following schema of equivalence:

$$\nu d.([d!/x]P_1 + [d!/x]P_2 \mid \delta(d?.Q)) \sim_\tau \nu d.([d!/x]P_1 \mid \delta(d?.Q)) + \nu d.([d!/x]P_2 \mid \delta(d?.Q)).$$

Up to symmetric cases there are two possible actions to consider. Also there is a correspondence between the actions and the reduced processes coincide.

[|] In the first place we observe the following law (*):

$$\nu d.([d!/x]P_1 \mid [d!/x]P_2 \mid \delta(d?.Q)) \sim_\tau \nu d.([d!/x]P_1 \mid \delta(d?.Q)) \mid \nu d.([d!/x]P_2 \mid \delta(d?.Q)).$$

This is shown by observing that the collection of terms with this shape forms a bisimulation. Up to symmetries we have three cases: (i) $[d!/x]P_1$ makes a non d! action, (ii) $[d!/x]P_1$ and $[d!/x]P_2$ communicate on a channel that is not d, (iii) $[d!/x]P_1$ makes a d! action and $\delta(d?.Q)$ a d? action. Next it is enough to develop the definition and apply the law (*):

$\{Q/x\}P_1 \mid P_2 \equiv \nu d.([d!/x]P_1 \mid [d!/x]P_2 \mid \delta(d?.Q))$
$[Q/x]_\tau P_1 \mid P_2 \equiv [Q/x]_\tau P_1 \mid [Q/x]_\tau P_2 \sim_\tau \{Q/x\}P_1 \mid \{Q/x\}P_2$ (by ind. hyp. and cong.).

[δ] $\{Q/x\}\delta P \equiv \nu d.(\delta([d!/x]P) \mid \delta(d?.Q))$.

$[Q/x]_\tau \delta P \equiv \delta[Q/x]_\tau P \sim_\tau \delta\{Q/x\}P \equiv \delta(\nu d.([d!/x]P \mid \delta(d?.Q)))$
(by ind. hyp. and cong.).

We prove the following schema of equivalence:

$\nu d.(\delta([d!/x]P) \mid \delta(d?.Q)) \sim \delta(\nu d.([d!/x]P \mid \delta(d?.Q)))$

There are two cases: (i) $[d!/x]P$ makes a non d! action, (ii) we perform an internal communication over the d channel. In both cases we reduce to the parallel composition of two processes such that we can apply the law (*) at one component whereas the other component can be matched back to the initial problem.

[v] Up to variable redenomination we have:
$\{Q/x\}\nu c.P \equiv \nu d.(\nu c.([d!/x]P) \mid \delta(d?.Q))$.
$[Q/x]_\tau \nu c.P \equiv \nu c.[Q/x]_\tau P \sim_\tau \nu c.\{Q/x\}P \equiv \nu c.(\nu d.([d!/x]P \mid \delta(d?.Q)))$
(by ind. hyp. and cong.).

[!] $\{Q/x\}c!P_1.\, P_2 \equiv \nu d.(c![d!/x]P_1.[d!/x]P_2 \mid \delta(d?.Q))$.
$[Q/x]_\tau c!P_1.\, P_2 \equiv c![Q/x]_\tau P_1.\, [Q/x]_\tau P_2 \sim_\tau c!\{Q/x\}P_1.\, \{Q/x\}P_2$.
Both processes make a c! action. After that we have to observe, say:
$\nu d.([d!/x]P_2 \mid \delta(d?.Q) \mid [\,[d!/x]P_1/z]_\tau R) \sim_\tau$ (by the (*) law)
$\nu d.([d!/x]P_2 \mid \delta(d?.Q)) \mid \nu d.(\delta(d?.Q) \mid [\,[d!/x]P_1/z]_\tau R) \equiv$
$\{Q/x\}P_2 \mid \{Q/x\}[P_1/z]_\tau R \sim_\tau \{Q/x\}P_2 \mid [\{Q/x\}P_1/z]_\tau R$ (by some analysis).

[?] $\{Q/x\}c?v.P \equiv \nu d.(c?v.[d!/x]P \mid \delta(d?.Q))$.
$[Q/x]_\tau c?v.P \equiv c?v.[Q/x]_\tau P \sim_\tau c?v.\{Q/x\}P$
By ind. hyp. and cong. for ?. Here we use the fact that in the conclusion the context is underlined. □

4.2.3 Lemma *(the τ-substitution simulation is preserved by the translation)*
If $\vdash \Gamma,\Gamma',x{:}Pr \supset P{:}\, Pr$, $\vdash \Gamma,\Gamma_1 \supset Q{:}\, Pr$, Γ,Γ',Γ_1 well-formed then
$\langle \underline{\Gamma,\Gamma',\Gamma_1} \rangle \supset \langle \{Q/x\}P \rangle \sim \langle [Q/x]_\tau P \rangle$.

Proof

Observe: $\langle [Q/x]_\tau P \rangle \equiv [\langle Q \rangle / x!]_\tau \langle P \rangle$, for appropriate contexts, and an obvious extension of the definition of τ-substitution. Also note:

$\langle \{Q/x\}P \rangle \sim \nu d.([d/x]\langle P \rangle \mid \delta(d?.\langle Q \rangle))$.

Hence proving $\langle [Q/x]_\tau P \rangle \sim \langle \{Q/x\}P \rangle$ is equivalent to prove:

$[\langle Q \rangle / x!]_\tau \langle P \rangle \sim \nu d.([d/x]\langle P \rangle \mid \delta(d?.\langle Q \rangle))$.

Next one proceeds by arguments similar to those in the previous lemma. □

4.3 Relating CH_τ and π-calculus actions

We now wish to relate the actions performed by a CH_τ process P and by its translation. There is a very good correspondence between the actions performed by P and <P> (this correspondence holds only at the first step) that is expressed by the two following lemmas. To ease the reading of the statement we take the convention that P, P',... denote CH_τ processes, and Q, Q',... denote π-processes. We delay the proofs to the appendix, they use in an essential way the substitution lemmas.

4.3.1 Lemma *(From CH_τ-actions to π-actions)*

If $\vdash \Gamma \supset P{:}Pr$ and $\vdash \Gamma \supset P \mapsto_\alpha \Gamma' \supset P'$ then:

(τ) $\alpha \equiv \tau$ implies there is $\vdash \langle\Gamma \supset P\rangle \mapsto_\tau \langle\Gamma\rangle \supset Q'$ and $\underline{\leq\Gamma>} \supset \langle P'\rangle \sim Q'$.

(?) $\alpha \equiv c?$, $\Gamma' \equiv \Gamma, x{:}Pr$ implies there is $\vdash \langle\Gamma \supset P\rangle \mapsto_{c?} \langle\Gamma\rangle, x{:}Ch \supset Q'$, and $\underline{\leq\Gamma>, x{:}Ch} \supset \langle P'\rangle \sim Q'$.

(!) $\alpha \equiv c!P_1$, $\Gamma' \equiv \Gamma, \Gamma_1$ implies there is $\vdash \langle\Gamma \supset P\rangle \mapsto_{c!d} \langle\Gamma\rangle, d{:}Ch \supset Q'$ and $\underline{\leq\Gamma>, d{:}Ch} \supset \nu\Gamma_1.(\delta(d?.\langle P_1\rangle) \mid \langle P'\rangle) \sim Q'$.

4.3.2 Lemma *(From π-actions to CH_τ-actions)*

If $\vdash \Gamma \supset P{:}Pr$, and there is $\vdash \Gamma \supset \langle P\rangle \mapsto_\alpha \langle\Gamma\rangle, \Gamma' \supset Q'$ then:

(τ) $\alpha \equiv \tau$ implies there is $\vdash \Gamma \supset P \mapsto_\tau \Gamma \supset P'$ and $\underline{\leq\Gamma>} \supset \langle P'\rangle \sim Q'$.

(?) $\alpha \equiv c?$, $\Gamma' \equiv \Gamma, x{:}Ch$ implies there is $\vdash \Gamma \supset P \mapsto_{c?} \Gamma, x{:}Pr \supset P'$ and $\underline{\leq\Gamma>, x{:}Ch} \supset \langle P'\rangle \sim Q'$.

(!) $\alpha \equiv c!d$ implies $\Gamma' \equiv d{:}Ch$, there is $\vdash \Gamma \supset P \mapsto_{c!P''} \Gamma, \Gamma_1 \supset P'$ and $\underline{\leq\Gamma>, d{:}Ch} \supset \nu\Gamma_1.(\delta(d?.\langle P''\rangle) \mid \langle P'\rangle) \sim Q'$.

(!') $\alpha \equiv x!$ implies $\Gamma' \equiv \emptyset$, $x{:}Pr \in \Gamma$, there is $\vdash \Gamma \supset [x!/x]P \mapsto_{x!} \Gamma \supset [x!/x]P'$ and $\underline{\leq\Gamma>} \supset \langle P'\rangle \sim Q'$.

4.4 Reduction of CH_τ bisimulation to π-calculus bisimulation

We are now ready to state the main result of this section showing that the equivalence of the translated processes implies the equivalence of the original CH_τ terms.

4.4.1 Theorem *(the translation is injective, up-to-bisimulations)*

If $\langle\Gamma\rangle \supset \langle P_1\rangle \sim \langle P_2\rangle$ then $\Gamma \supset P_1 \sim_\tau P_2$.

Proof

In the following we omit the contexts that are not strictly relevant to the comprehension of the proof. The indices "(i)" in the diagrams show the flow of the reasoning. The rules by which we go back and forth are fixed by the previous two lemmas. We show:

$$S_{CH}(\Gamma) \triangleq \{ (P_1, P_2) \mid \langle\Gamma\rangle \supset \langle P_1\rangle \sim \langle P_2\rangle \}$$

is a τ-sgb bisimulation. Hence:

$<\Gamma>\supset <P_1> \sim <P_2> \Rightarrow \Gamma\supset P_1 S_{CH} P_2 \Rightarrow \Gamma \supset P_1 \sim_\tau P_2$.

Suppose: $P_1\ S_{CH}\ P_2$, $P_1 \mapsto_\alpha P_1'$.

- $\alpha \equiv \tau$.
$P_1\ S_{CH}\ P_2$; $<P_1> \sim <P_2>$
(1) $\tau\downarrow$ $\tau\downarrow$ (4) ; (2) $\tau\downarrow$ $\tau\downarrow$ (3)
$P_1'\ S_{CH}\ P_2'$; $<P_1'> \sim Q_1' \sim Q_2' \sim <P_2'>$

- $\alpha \equiv c?$.
$P_1\ S_{CH}\ P_2$; $<P_1> \sim <P_2>$
(1) $c?\downarrow$ $c?\downarrow$ (4) ; (2) $c?\downarrow$ $c?\downarrow$ (3)
$\underline{x}\supset P_1'\ S_{CH}\ P_2'$; $\underline{x}\supset<P_1'>\sim Q_1' \sim Q_2'\sim<P_2'>\subset\underline{x}$

- $\alpha \equiv c!d$.
$P_1\ S_{CH}\ P_2$; $<P_1> \sim <P_2>$
(1) $c!P_1''\downarrow$ $c!P_2''\downarrow$ (4) ; (2) $c!d\downarrow$ $c!d\downarrow$ (3)
$\Gamma_1\supset P_1'$ $\Gamma_2\supset P_2'$; $d\supset Q_1' \sim Q_2'$

where: $d\supset \nu\Gamma_1.(<P_1'> \mid \delta(d?<P_1''>) \sim Q_1'$, $d\supset \nu\Gamma_2.(<P_2'> \mid \delta(d?<P_2''>) \sim Q_2'$.
We have to show: $\nu\Gamma_1.(P_1' \mid [P_1''/x]_\tau R)\ S_{CH}\ \nu\Gamma_2.(P_2' \mid [P_2''/x]_\tau R)$, any R . Observe:
(i) $[P_i''/x]_\tau R \sim_\tau \{P_i''/x\}R \equiv \nu d.([d!/x]R \mid \delta(d?P_i'')\ )$, i=1,2.
(ii) $\nu d.(Q_i' \mid [d/x]<R>) \sim <\nu\Gamma_i.(P_i' \mid [P_i''/x]_\tau R)>$, i=1,2. □

Example: We consider the τ-variant of the equivalence discussed in section 3:

$P_1 \equiv \nu c.\nu d.a!(d?.c?.\emptyset).c!.P' \sim_\tau a!\emptyset.\emptyset \equiv P_2$.

We can prove it via the translation by observing:

$<P_1> \sim \nu c.\nu d.\nu d'.(a!d'.c!.<P'> \mid \delta(d'?.(d?.c?.\emptyset))\ ) \mapsto_{a!d'}$
$d'\supset \nu c.\nu d.(c!.<P'> \mid \delta(d'?.(d?.c?.\emptyset))\ ) \equiv Q_1$
$<P_2> \sim \nu d'.(a!d' \mid \delta(d'?.\emptyset)\ ) \mapsto_{a!d'} d'\supset\emptyset \mid \delta(d'?.\emptyset) \equiv Q_2$.

It is then easy to show: $Q_1 \sim Q_2$, by building a suitable bisimulation containing all pairs: $\nu c.(c!.<P'> \mid \nu d.(d?.c? \mid ... \mid d?.c? \mid \delta(d'?.d?.c?)\ ))$, $\delta(d'?)$. □

Note *(on preservation)*: It is known (Thomsen[89(a)]) that:
if $\underline{\Gamma} \supset P_1 \sim_{CH,\tau} P_2$ then $<\Gamma>\supset <P_1> \sim <P_2>$. (1)
where $\sim_{CH,\tau}$ is $\sim_{CH}$ as defined in section 3, adapted to the CH_τ calculus. The extension of (1) to $\sim_\tau$ is an open problem. □

Remark *(CH vs. CH_τ)*

The following examples suggest that there are problems in relating the bisimulation of CH and CHτ.

(1) $P \equiv \nu c.(c!(a!) \mid c?x.(x+b!)) \sim P' \equiv \nu c.(c!\emptyset \mid c?x.(a!+b!)) \sim \tau.(a!+b!)$, but $P \sim_\tau \tau.(\tau.a!+b!)$ that is not even weakly bisimilar to: $\tau.(a!+b!) \sim_\tau P'$.

(2) Symmetrically take $P \equiv \nu c.(c!(a!) \mid c?x.(x+b!)) \sim_\tau P' \equiv \nu c.(c!\emptyset \mid c?x.(\tau.a!+b!)) \sim_\tau \tau.(\tau.a!+b!)$, but $P \sim \tau.(a!+b!)$ is not even weakly bisimilar to $P' \sim \tau.(\tau.a!+b!)$.

The point of this examples is that CH is a higher level calculus than CHτ in which

we abstract away from the τ-actions needed to actually perform a process communication. In certain situations the extra τ-actions are a nuisance, as it is well known that weak bisimulation is sensitive to the the presence of τ's. However we expect that one can set reasonable conditions on processes so that the introduction of τ-actions does not alter the equivalence. We leave this problem to further investigation. □

5. Bisimulation for π-calculus Revisited

From the viewpoint of automatic verification the current formulation of (strong) bisimulation for, say, the π-calculus has two (related) main drawbacks: (i) in the case of an input action in order to check the bisimilarity of two processes we have to consider all possible inputs, (ii) the definition of sb is built over the notion of sgb by roughly quantifying over all possible substitutions.

More generally the fact that variables have not a first grade status but are always treated via a reduction to all closed instances is rather unsatisfying. In this section we make an attempt for placing directly the notion of variable in the theory. Certainly we cannot claim any final answer, however this attempt does produce a notion of strong *uniform* bisimulation that is contained in the notion of strong bisimulation and appears closer to a reasonable verification system for the π-calculus.

5.1 Substitutions

In the first place we reformulate the notion of strong bisimulation for the π-calculus in a way that makes explicit the *bounded* nature of two universal quantifications used in its definition. Let us start with two obvious remarks:

- In establishing a s(g)b, say $\Delta \supset P \sim Q$, elements in the context that do not appear in P and Q are irrelevant.

- If σ is an injective substitution then the transition of the judgment $\Delta \supset P$ are in perfect correspondence, up-to-redenomination, with the transitions of the judgment $\sigma\Delta \supset \sigma P$. Similarly proving $\Delta \supset P \sim Q$ is the "same" as proving $\sigma\Delta \supset \sigma P \sim \sigma Q$.

In other words it is not the name of the syntactic objects that counts but only their *distinctions* (this point is also stressed in Milner&al[89]). It follows that in the verification of $\Delta \supset P \sim Q$ there are only a *finite* number of substitutions to check. We now quantify this *finite* a bit more precisely.

Conventions: Given a context Δ, we say that two substitutions σ, σ' are Δ-*equivalent* when they make the same distinctions over Δ, i.e. $\sigma =_\Delta \sigma'$ iff $\forall x, y \in \Delta.$ $(\sigma(x) = \sigma(y) \Leftrightarrow \sigma'(x) = \sigma'(y))$. Given a context Δ and variables x, y occurring in Δ we write $x <_\Delta y$ ($x \leq_\Delta y$) if x occurs before y in Δ (or x=y). By $'\Delta$ we mean the context Δ where all underlines have been removed.

5.1.1 Definition *(Δ-substitution)*

Given a context Δ containing names $\{x_1, ..., x_n\}$ we say that a substitution σ is a(n admissible) *Δ-substitution* if: (i) σ is a retraction over Δ, i.e. $\sigma_{|\Delta} = \sigma \circ \sigma_{|\Delta}$, (ii) $x{:}Ch \in \Delta$ implies $\sigma(x) = x$, (iii) $\underline{x{:}Ch} \in \Delta$ implies $\sigma(x) \leq_\Delta x$.

5.1.2 Proposition *(Δ-substitutions are canonical and enough)*

(1) Let Δ be a context and σ, σ' Δ-substitutions. If $\sigma \simeq_\Delta \sigma'$ then $\sigma = \sigma'$.

(2) Let Γ, $\underline{\Gamma'}$ be a context and σ a substitution s.t. $\sigma(c) = c$ for $c \in \Gamma$. Then there is a Γ, $\underline{\Gamma'}$-substitution σ' such that $\sigma \simeq_{\Gamma, \underline{\Gamma'}} \sigma'$.

Proof

(1) First observe that if σ is a Δ, Δ'-substitution then $\sigma_{|\Delta}$ is a Δ-substitution. Next proceed by induction on the size of Δ. The base case is trivial. For the induction step the interesting case is when we have $\Delta' \equiv \Delta, \underline{x{:}Ch}$. Now consider the behavior of σ, σ' on x: (i) $\sigma(x) = \sigma'(x) = x$, we are done. (ii) $\sigma(x) <_{\Delta'} \sigma'(x) = x$. But then: $\sigma(\sigma(x)) = \sigma(x)$, whereas $\sigma'(\sigma(x)) <_{\Delta'} \sigma'(x) = x$. (iii) $\sigma(x) <_{\Delta'} x$ and $\sigma'(x) <_{\Delta'} x$. By ind. hyp. on Δ: $\sigma'(\sigma(x)) = \sigma(\sigma(x)) = \sigma(x)$, and symmetrically $\sigma(\sigma'(x)) = \sigma'(\sigma'(x)) = \sigma'(x)$. Now if $\sigma(x) \neq \sigma'(x)$ then $\sigma'(x) = \sigma'(\sigma'(x))$ whereas $\sigma(x) \neq \sigma(\sigma'(x)) = \sigma'(x)$.

(2) Define an equivalence on elements in Γ, $\underline{\Gamma'}$ as $x \cong y$ iff $\sigma(x) = \sigma(y)$. Next define $\sigma'(x) = \min[x]_\cong$, where $[x]_\cong$ is the equivalence class of x and the order is the one given by Γ, $\underline{\Gamma'}$. It is now easy to verify that σ' is the Γ, $\underline{\Gamma'}$-substitution we looked for. Note that (2) fails if we allow to put a variable before a constant as in: $\underline{x{:}Ch}$, c:Ch. □

Observe that when Δ-substituting we can just keep the same context, that is:

$$\vdash \Delta \supset P{:}Pr, \quad \sigma \text{ } \Delta\text{-substitution} \Rightarrow \vdash \Delta \supset \sigma P{:}Pr.$$

Having found a canonical way to represent substitutions up-to-equivalence we can now restate the *notion of sb for the π-calculus* as follows:

$$\Gamma, \underline{\Gamma'} \supset P \sim Q \text{ iff for all } \sigma \text{ } \Gamma, \underline{\Gamma'}\text{-substitution } \Gamma, \Gamma' \supset \sigma P \sim \sigma Q$$

The same idea also entails the following reformulation of the *(?) clause of the definition of sgb for the π-calculus.*

(?) $\ldots \vdash \Gamma \supset P \mapsto_d ? \Gamma, x{:}Ch \supset P'$ implies there is
$\vdash \Gamma \supset Q \mapsto_d ? \Gamma, x{:}Ch \supset Q'$, for all σ Γ, $\underline{x{:}Ch}$-substitution Γ, $x{:}Ch \supset \sigma P' \sim \sigma Q'$.

Supposing to have eliminated all redundant variables from Γ, $\underline{\Gamma'}$ we can now observe that the number of checks to be made *grows exponentially* in the size of $\underline{\Gamma'}$ for the definition of sb, and it amounts to n+1 if n is the size of Γ for the clause (?). We expect that a control on the size of the checks to be made is critical in the implementation of a practical verification tool. We now formalize a transition system that takes into account the notion of variable. On the pragmatic side this approach may loosely resemble a lazy evaluation in that substitutions are not carried on till the moment they can actually have an impact on the transitions.

5.2 A new labelled transition system

The binders ν and ? are of a very different nature: the former has the effect of creating a new name, whereas the latter creates a place-holder that can be instantiated with an arbitrary value. We wish to remember in the context, by which binder the assumption was created. For this purpose it is enough to put a marker on the assumption created by the (?) rule. Henceforth using a convention that matches the one introduced in section 3 we *underline* such an assumption as follows:

$$(?)_M \qquad \frac{}{\Delta \supset (c?v.\ P) \mapsto c?\ \Delta, \underline{v{:}Ch} \supset P}$$

Next let us re-examine the communication rule:

$$(|.com!?) \qquad \frac{\Gamma \supset P \mapsto x!d\ \ \Gamma,\Gamma' \supset P' \qquad \Gamma \supset Q \mapsto y?\ \Gamma, v{:}Ch \supset Q' \qquad (x \equiv y)}{\Gamma \supset (P \mid Q) \mapsto \tau\ \Gamma \supset \nu\Gamma'.(P' \mid [d/v]Q')}$$

The idea is to weaken the hypothesis $(x \equiv y)$ by allowing communication also when x and y "match" according to certain rules. Given a context Δ and variables x, y occurring in Δ we write: $x \in ?(\Delta)$ if x is underlined, and $x \in cst(\Delta)$ if x is not underlined. The intuition that leads to the matching condition is outlined by the following reasoning. Suppose $x <_\Delta y$, then:

(i) if $y \in ?(\Delta)$ then y is just a place holder for some channel known by the environment *before* the communication takes place. Among such channels we find x (as $x <_\Delta y$), hence we may identify y with x and perform a communication.

(ii) if $y \in cst(\Delta)$ then y is a constant name that cannot be identified with a previous name.

Observe that the *order of assumptions here is important*. For instance, as a corollary of (ii), if $x \in ?(\Delta)$ and $y \in cst(\Delta)$ then x *cannot* match y. The reason is that the name y was created *after* the input action on x has taken place.

We can formalize quite nicely the new communication rule via the following function *Match(x, y, Δ)* that returns a truth-value and a substitution:

$$\text{Match}(x, y, \Delta) \triangleq \begin{cases} (\text{true}, \text{id}) & \text{if } x \equiv y \\ (\text{true}, [x/y]) & \text{if } x <_\Delta y,\ y \in ?(\Delta) \\ (\text{true}, [y/x]) & \text{if } y <_\Delta x,\ x \in ?(\Delta) \\ (\text{false}, \text{id}) & \text{o.w.} \end{cases}$$

$$(|.com!?)_M \qquad \frac{\Delta \supset P \mapsto x!d\ \ \Delta,\Delta' \supset P' \qquad \Delta \supset Q \mapsto y?\ \Delta, \underline{v{:}Ch} \supset Q' \qquad \text{Match}(x, y, \Delta) = (\text{true}, \sigma)}{\Delta \supset (P \mid Q) \mapsto (\tau, \sigma)\ \Delta \supset \nu\Delta'.(P' \mid [d/v]Q')}$$

The rule $(|.com?!)_M$ is written symmetrically. The rule $(|.com!?)_M$ says that whenever there are two dual communications on two matching channels one can

perform a τ-action along with a substitution σ generated by the match. We denote with lts_M the labelled transition system (M being a mnemonic for the match condition) that coincides with the one defined in section 2 but for the following exceptions: the rules (?) and (!.com!?) are replaced, respectively, by $(?)_M$ and $(!.com!?)_M$, (ii) the contexts Γ are replaced by the more general contexts Δ. The two following lemmas make precise the connections between lts and lts_M (we omit to remove the underlines in a context when working with the lts of section 3). Their proofs proceed by induction on the length of the transition derivation and they are delayed to the appendix.

5.2.1 Lemma *(from lts to lts_M transitions)*

Suppose that $\vdash\Delta\supset P$: Pr, and σ is a Δ-substitution. If $\vdash \Delta\supset\sigma P\mapsto\alpha\, \Delta,\Delta_1\supset P_1$ then there is $\vdash_M \Delta\supset P\mapsto\alpha'\ \Delta,\Delta_1'\supset P_1'$, s.t. $\sigma P_1' \equiv P_1$ and

- $\alpha \equiv \tau$: $\alpha' \equiv (\tau, \sigma')$, $\sigma \circ \sigma' = \sigma$.
- $\alpha \equiv d?$, $\Delta_1 \equiv x{:}Ch$: $\alpha' \equiv d'?$, $\sigma(d') = d$, $\Delta_1' \equiv \underline{x{:}Ch} \equiv \underline{\Delta_1}$.
- $\alpha \equiv d!c$: $\alpha' \equiv d'!c'$, $\sigma(d') = d$, $\sigma(c') = c$, $\Delta_1' \equiv \Delta_1$.

5.2.2 Lemma *(from lts_M to lts transitions)*

Suppose that $\vdash\Delta\supset P$: Pr, and σ is a Δ-substitution. If $\vdash_M \Delta\supset P\mapsto\alpha\, \Delta,\Delta'\supset P'$ then

- $\alpha \equiv (\tau, \sigma)$: there is $\vdash \Delta\supset\sigma P\mapsto\tau\, \Delta\supset\sigma P'$.
- o.w.: there is $\vdash \Delta\supset P\mapsto\alpha\, \Delta,\Delta'\supset P'$ (with Δ' not underlined).

5.3 Uniform Strong Bisimulation

We now introduce the notion of strong bisimulation naturally associated to lts_M, we will denote it with sb_u. The 'u' stands for *uniform* and, as mentioned in the introduction, it indicates a peculiar character of this bisimulation w.r.t. substitutions.

5.3.1 Definition *(uniform strong bisimulation for π-calculus)*

A family of relations S is a sb_u if $\Delta \supset P\ S\ Q$ and for any $\vdash_M \Delta\supset P \mapsto\alpha\, \Delta'\supset P'$:

(τ) $\alpha \equiv (\tau, \sigma)$ implies there is $\vdash_M \Delta \supset Q \mapsto(\tau, \sigma) \Delta \supset Q'$, $\Delta \supset \sigma P' S \sigma Q'$.

(?) $\alpha \equiv d?$, $\Delta' \equiv \Delta,\underline{x{:}Ch}$, implies there is $\vdash_M \Delta\supset Q \mapsto d?\, \Delta,\underline{x{:}Ch}\supset Q'$, $\Delta,\underline{x{:}Ch}\supset P'SQ'$.

(!) $\alpha \equiv d!c$ implies ($\Delta'\equiv\Delta,c{:}Ch$ or $\Delta'\equiv\Delta$), there is $\vdash_M \Delta\supset Q \mapsto d!c\ \Delta'\supset Q'$, $\Delta'\supset P'\ S\ Q'$.

and symmetrically. As usual we write $\Delta \supset P \sim_u Q$ if $\Delta \supset P\ S\ Q$ for some S sb_u.

5.3.2 Theorem *(sb_u is contained in sb)*

In the fragment corresponding to the π-calculus

if $\Gamma, \underline{\Gamma'} \supset P \sim_u Q$ then $\Gamma, \underline{\Gamma'} \supset P \sim Q$.

Proof

We define S_u as the least family of relations that satisfies:

$\Delta \supset P \sim_u Q$, σ Δ-substitution $\Rightarrow$ '$\Delta\supset \sigma P\ S_u\ \sigma Q$.

Observe that if S_u is a sgb then we can conclude as follows:

$\Gamma, \underline{\Gamma}' \supset P \sim_u Q \Rightarrow_{(1)} \Gamma, \Gamma' \supset \sigma P\, S_u\, \sigma Q$, for any σ $\Gamma, \underline{\Gamma}'$-sub. $\Rightarrow_{(2)}$
$\Gamma, \Gamma' \supset \sigma P \sim \sigma Q$, for any σ $\Gamma, \underline{\Gamma}'$-sub. $\Rightarrow_{(3)}$ $\Gamma, \underline{\Gamma}' \supset P \sim Q$.

(1) By definition of S_u. (2) By definition of $\sim$. (3) By the revised definition of sb. Hence it remains to show S_u is a sgb. Suppose:

$\Delta \supset P \sim_u Q$, σ Δ-sub., '$\Delta \supset \sigma P\, S_u\, \sigma Q$, $\vdash \Delta \supset \sigma P \mapsto^{\alpha} \Delta, \Delta_1 \supset P_1$. By lemma 5.2.1 there is: $\vdash_M \Delta \supset P \mapsto^{\alpha'} \Delta, \Delta_1' \supset P_1'$, s.t. $\sigma P_1' \equiv P_1$, and

- $\alpha \equiv \tau$: $\alpha' \equiv (\tau, \sigma')$, $\sigma \circ \sigma' = \sigma \Rightarrow \vdash_M \Delta \supset Q \mapsto^{(\tau, \sigma')} \Delta \supset Q_1'$, $\Delta \supset \sigma' P_1' \sim_u \sigma' Q_1' \Rightarrow$
  $\vdash \Delta \supset \sigma Q \mapsto^{\tau} \Delta \supset \sigma Q_1'$, $\Delta \supset \sigma P_1'\, S_u\, \sigma Q_1'$.
- $\alpha \equiv d?$, $\Delta_1 \equiv x{:}Ch$, $\alpha' \equiv d'?$, $\sigma(d') = d$, $\Delta_1' \equiv \underline{x{:}Ch} \Rightarrow \vdash_M \Delta \supset Q \mapsto^{d'?} \Delta, \underline{x{:}Ch} \supset Q_1'$, and
  $\Delta, \underline{x{:}Ch} \supset P_1' \sim_u Q_1' \Rightarrow \vdash \Delta \supset \sigma Q \mapsto^{d?} \Delta, x{:}Ch \supset \sigma Q_1'$, '$\Delta, \underline{x{:}Ch} \supset \sigma P_1'\, S_u\, \sigma Q_1'$.
- $\alpha \equiv d!c$: $\alpha' \equiv d'!c'$, $\sigma(d') = d$, $\sigma(c') = c$, $\Delta_1' \equiv \Delta_1 \Rightarrow \vdash_M \Delta \supset Q \mapsto^{d'!c'} \Delta, \Delta_1 \supset Q_1'$, and
  $\Delta, \Delta_1 \supset P_1' \sim_u Q_1' \Rightarrow \vdash \Delta \supset \sigma Q \mapsto^{d!c} \Delta, \Delta_1 \supset \sigma Q_1'$, '$\Delta, \Delta_1 \supset \sigma P_1'\, S_u\, \sigma Q_1'$.

Note: basically the same proof with $\sim_u$ at the place of S_u shows that sb_u is closed under Δ-substitutions, i.e. $\Delta \supset P \sim_u Q$, σ Δ-substitution $\Rightarrow \Delta \supset \sigma P \sim_u \sigma Q$. □

We now explain, in an informal notation, the uniformity of sb_u. Suppose $P \sim_u Q$, then we know $\sigma P \sim \sigma Q$, for any σ. However we can say something more about the relationship between transitions and substitutions! Namely suppose $\vdash_M P \mapsto^{\alpha} P'$, then we can find Q' s.t. $\vdash_M Q \mapsto^{\alpha} Q'$ and $P' \sim_u Q'$, that is we can find a Q' s.t. for any σ $\sigma P' \sim \sigma Q'$. *It is this ability to select a transition that simulates for any substitution that we call uniformity.* Unfortunately sb does not enjoy this property as shown by the following

Example *(sb_u is strictly contained in sb)*

This example can be carried on in a CCS with variables. We work in the context $\Delta \equiv x{:}Ch, \underline{y{:}Ch}$, hence y is a variable that can match the constant x. The two Δ-substitutions are id and [x/y]. Define:

$n \equiv x!.y? + y?.x!$	(it *never* makes a τ-action)
$s \equiv x! \mid y?$	(sometime it makes a τ-action)
$a \equiv x! \mid y? + \tau$	(it *always* makes a τ-action)

We build two processes P, Q such that $\Delta \supset P \sim Q$, but *not* $\Delta \supset P \sim_u Q$, as follows:

$P \equiv x!.n + x!.s + x!.a$ $\qquad$ $Q \equiv x!.n + x!.a$

Now observe that sgb partitions the set {n, s, a} as follows depending on Δ-sub. :

for id:	{n, s}, {a}
for [x/y]:	{n}, {s, a}

Note that s in P is covered by n and a, at the varying of the Δ-substitution. This suffice to show $\Delta \supset P \sim Q$. On the other hand we have: $\vdash_M \Delta \supset P \mapsto^{x!} \Delta \supset$ s. But neither n nor a are uniformly bisimilar to s. Hence Q *cannot* (uniformly) simulate this P transition. We leave to further investigation the problem of determining a

class of processes for which sb_u coincides with sb. □

Conclusion

There is an obvious need to consider more flexible notions of equivalence than that of strong bisimulation. It appears that all definitions can be easily adapted to the notion of weak bisimulation, however their relationships remain to be investigated. One may also wish to distinguish behaviors such as termination, deadlock, and divergence (see, e.g., Aceto&Hennessy[92]). Concerning termination it has been observed by B. Thomsen that CHOCS *without duplication* can already simulate full recursive definitions. It would be interesting to devise a "type system" that guarantees termination of a large class of CHOCS programs

The careful presentation of the calculus *with contexts* given here can be seen as a first step towards the definition of variable free calculi (following DeBrujin notation), and of abstract 'categorical' interpretations, say in the style of Winskel[90]. The latter problem should also bring to the limelight the analysis of the "type structure" of such process calculi.

Some practical evidence needs to be gathered about the utility of the translation from CHOCS into π-calculus, and the notion of uniform strong bisimulation.

As for applications we refer, for instance, to Aït-Mohamed&Amadio[92] where we give a π-calculus simulation of an environment machine for lazy evaluation.

References

Aceto L., Hennessy M. [1992] "Termination, deadlock, and divergence", JACM, 39, (147-187).

Aït-Mohamed O., Amadio R. [1992] "A simulation of an environment machine in π-calculus", preprint INRIA-Lorraine.

Astesiano E., Zucca E. [1984] "Parametric channels via Label Expressions in CCS", TCS, 33, (45-64).

Boudol G. [1989] "Towards a lambda calculus for concurrent and communicating systems", in Proc. TAPSOFT, SLNCS 351.

Engberg U., Nielsen M. [1986] "A calculus of communicating systems with label passing", DAIMI PB-208, CS Dept. Aarhus.

Milner R. [1989] "Communication and Concurrency", Prentice Hall.

Milner R., Parrow J., Walker D. [1989] "A calculus of mobile processes I, II", ECS-LFCS TR 89-85, 89-86, Edinburgh.

Nielsen M. [1988] "The typed lambda calculus with first class processes", TR 1988-43, Inst. for Datateknik, Lingby.

Pratt T.W.[1984] "Programming Languages. Design and Implementation", Prentice-Hall, 2nd edition.

Thomsen B. [1989] "A calculus of higher order communicating systems", in Proc. ACM-POPL, 1989.

Thomsen B. [1989(a)] "Plain CHOCS", manuscript Imperial College, London, 1989.

Winskel G. [1990] "A compositional proof system on a category of labelled transition systems", Info&Comp, 87, 3, (2-57).

Nancy, Sat, Jul 4, 1992

Appendix

In this appendix we collect some of the proofs omitted in the paper.

Proofs section 1

1.2.1 Proposition

(1) *Exchange.* If $\vdash\Gamma \supset \text{exp: K}$ then $\vdash\sigma(\Gamma) \supset \text{exp: K}$.

(2) *Weakening.* If $\vdash\Gamma \supset \text{exp: K}$ and $\Gamma\cap\Gamma' = \varnothing$ then $\vdash\Gamma,\Gamma' \supset \text{exp: K}$.

(3) *Removing Variables.* If $\vdash\Gamma,v{:}K \supset \text{exp: K}$ and $v \notin \text{exp}$ then $\vdash\Gamma \supset \text{exp: K}$.

(4) *Substitution.* If $\vdash\Gamma,\Gamma',v{:}K_1 \supset \text{exp: K}$, $\vdash\Gamma,\Gamma_1 \supset \text{exp}_1\text{: } K_1$, Γ,Γ',Γ_1 well-formed then $\vdash\Gamma,\Gamma',\Gamma_1 \supset [\text{exp}_1/v]\ \text{exp: K}$.

Proof

All the following proofs are by induction on the length of the derivation of the kind judgment in the hypothesis. In the following we just consider the rules (asmp), (+), and (v) as the rule (ø) is always trivial, the rules (|), (δ), and (!) behave like (+), and the rule (?) behaves like (v).

(1) Observe that if Γ is a well-formed context then so is $\sigma(\Gamma)$.

(asmp) If $v{:}K \in \Gamma$ then $v{:}K \in \sigma(\Gamma)$ and apply (asmp).

(+) Use ind. hyp. on the premises of the rule and apply (+).

(v) Use ind. hyp. on the premiss of the rule, consider an appropriate substitution, and apply (v).

(2) Observe that if $\Gamma\cap\Gamma' = \varnothing$ then Γ,Γ' is well formed.

(asmp) If $v{:}K \in \Gamma$ then $v{:}K \in \Gamma,\Gamma'$ and apply (asmp).

(+) Use ind. hyp. on the premises of the rule and apply (+).

(v) Choose the variable bound by v so that it does not interfere with Γ'. Use inductive hypothesis on the premiss, transform the judgment by exchange (1), and then apply (v).

(3) Observe that Γ is a well-formed context.

(asmp) If $x{:}K \in \Gamma,v{:}K$ and $x \neq v$ then $x{:}K \in \Gamma$.

(+) If $v \notin (P+Q)$ then $v \notin P$ and $v \notin Q$. Use ind. hyp. on the premises of the rule and apply (+).

(v) Choose appropriately the variable bound by v. Apply the ind. hyp. on the premiss of the rule, if necessary transform the judgment by exchange (1), and then apply (v).

(4) (asmp) Suppose $x{:}K \in \Gamma,\Gamma',v{:}K$. There are two cases to consider: (i) if x=v then apply weakening (2) and exchange (1), (ii) o.w. remove variables (3) and apply weakening.

(+) Use ind. hyp. on the premises of the rule and apply (+).

(v) Choose appropriately the variable bound by v. Apply the ind. hyp. on the premiss of the rule, then use exchange (1), and apply (v). □

Proofs section 2

2.2 Proposition

(1) If $\vdash \Gamma \supset P \mapsto\alpha \Gamma' \supset P'$ then $\Gamma \subseteq \Gamma'$, and moreover
if $\alpha \equiv \tau$ then $\Gamma \equiv \Gamma'$.
if $\alpha \equiv d?$ then $\Gamma, x{:}Tv \equiv \Gamma'$ for some x.
if $\alpha \equiv d!t$ then $\Gamma, c_1{:}\,Ch, ..., c_m{:}\,Ch \equiv \Gamma'$, for some $m \geq 0$, $c_1, ..., c_m$.

(2) If $\vdash \Gamma \supset P{:}\,Pr$ and $\vdash \Gamma \supset P \mapsto\alpha \Gamma,\Gamma' \supset P'$ then
(a) $\vdash \Gamma,\Gamma' \supset P'{:}\,Pr$, and
(b) if $\alpha \equiv d!t$ then $\vdash \Gamma,\Gamma' \supset t{:}\,Tv$.

Proof

All the following proofs are by induction on the length of derivation of a judgment in the lts. The cases (+.right), (|.left), (|.right), (δ) behave as (+.left). Also (|.com?!) behaves like (|.com!?).

(1) This is shown by a simple inspection of the rules.

(2) (!) Use the premises of the kind rule (!).

(?) Use the premises of the kind rule (?).

(+.left) Use the premises of the kind rule (+) and the premises of the lts rule (+.left) to apply the inductive hypothesis .

(ν) Use the premises of the kind rule (ν) and the premiss of the lts rule (ν) to apply ind. hyp. Conclude using exchange and (ν) for the process, and remove variables for the action.

(ν.open) Use the premises of the kind rule (ν) and the premiss of the lts rule (ν.open) to apply ind. hyp.

(|.com!?) Use the premises of the kind rule (|) and inductive hypothesis to conclude: $\vdash \Gamma,\Gamma'' \supset P'{:}\,Pr$, $\vdash \Gamma, v{:}\,Tv \supset Q'{:}\,Pr$, and $\vdash \Gamma,\Gamma'' \supset t{:}\,Tv$. Use substitution to prove: $\vdash \Gamma,\Gamma'' \supset [t/v]Q'{:}\,Pr$, and then apply (ν). □

Proofs section 3

3.4.3 Lemma

If $\vdash \Gamma \supset P{:}\,Pr$, $\vdash \Gamma \supset Q{:}\,Pr$, $\Gamma,\Gamma'' \supset \nu c.(P \mid [c/x]R) \sim \nu c.(Q \mid [c/x]R)$ any $\vdash \Gamma,\Gamma'', x{:}\,Ch \supset R{:}\,Pr$, then $\Gamma, c{:}\,Ch \supset P \sim Q$.

Proof

We introduce a series of definitions and suitably modify and generalize the thesis. We suppose that a, $\{an\}_{n\in\omega}$, $\{bn\}_{n\in\omega}$, $\{dnm\}_{n,m\in\omega}$ are sequences of distinct fresh names. We abbreviate a list of names $c_1,..., c_m$ or a list of kind declarations $c_1{:}\,Ch..., c_m{:}\,Ch$ as **c**. We define parametrically in n, **c** terms R(n,**c**), R'(n, **c**) as follows:[9]

[9] We suggest carrying on the proof for the CCS fragment first (the definitions are much simpler).

$R(n,\mathbf{c}) ::=$ $\quad \emptyset \mid$

$\quad c!d.(R(n+1, \mathbf{c}) + an!) \mid$ (for some d not among the fresh names)

$\quad c?x.(R(n+1, \mathbf{c}) + an! + x!bn!) \mid$

$\quad c?x.(R(n+1, \mathbf{c}) + an! + Test(n, \mathbf{c}, x)) \mid$

$\quad c?x.(R(n+1, \mathbf{c}, x) + an! + x!bn! + Test(n, \mathbf{c}, x)) \mid$

$\quad a!c_1 ... a!c_m$

$Test(n, \mathbf{c}, x) \equiv (c_1!.d_{n1}! \mid x?.b_n!) + ... + (c_m!.d_{nm}! \mid x?.b_n!)$

$R'(n+1,\mathbf{c}) ::= \emptyset \mid an! \mid an! + x!bn! \mid$

$\quad an! + Test(n, \mathbf{c}, \mathbf{c}', x) \mid an! + x!bn! + Test(n, \mathbf{c}, \mathbf{c}', x).$ (x in $\mathbf{c}, \mathbf{c}'$)

Given $\vdash \Gamma, \mathbf{c} \supset P:Pr, \vdash \Gamma, \mathbf{c} \supset Q:Pr$ we set: $\Gamma, \mathbf{c} \supset PSQ$ if:

there is n, any R(n,c), there is R'(n,c) such that:
$\Gamma,\Gamma'' \supset \nu\mathbf{c}.(P \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \sim \nu\mathbf{c}.(Q \mid R(n,\mathbf{c})+R'(n,\mathbf{c}))$.

We claim that S is a π-sgb (*). In this case we prove our original statement as follows:

$\Gamma,\Gamma'' \supset \nu\mathbf{c}.(P \mid [c/x]R) \sim \nu\mathbf{c}.(Q \mid [c/x]R)$ any $\vdash \Gamma,\Gamma'',x:Ch \supset R:Pr$, implies
$\Gamma,\Gamma'' \supset \nu\mathbf{c}.(P \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \sim \nu\mathbf{c}.(Q \mid R(n,\mathbf{c})+R'(n,\mathbf{c}))$ any n, R(n,c), R'(n,c) implies
$\Gamma,c:Ch \supset PSQ$, implies
$\Gamma,c:Ch \supset P \sim \pi Q$ implies by 3.4.1
$\Gamma,c:Ch \supset P \sim Q$.

We now proceed with the proof of our claim (*). We feel free to omit contexts. We preserve the property that the fresh names do not occur in derivatives of P and Q. Suppose for some n, for any R(n,c), there is R'(n,c)

$\Gamma,\Gamma'' \supset \nu\mathbf{c}.(P \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \sim \nu\mathbf{c}.(Q \mid R(n,\mathbf{c})+R'(n,\mathbf{c}))$, and
$\vdash \Gamma,\mathbf{c} \supset P \mapsto\alpha \Gamma,\mathbf{c},\Gamma' \supset P'$.

We proceed by an analysis of the action α. The basic schema is that:

(i) an action on P forces
(ii) a corresponding action on $\nu\mathbf{c}.(P \mid R(n,\mathbf{c})+R'(n,\mathbf{c}))$, which forces
(iii) a corresponding action on $\nu\mathbf{c}.(Q \mid R(n,\mathbf{c})+R'(n,\mathbf{c}))$, generated by
(iv) an action on Q.

The construction of R(n,c) has been devised exactly to 'force the corresponding actions'. The fresh names are needed to signal to the external world that a certain internal transition has taken place. On the other hand R'(n,c) represents some residual of a previous interaction that we have to keep around.

- $\alpha \equiv \tau$, Γ' empty.

(i) If $P \mapsto\tau P'$, then
(ii) $\nu\mathbf{c}.(P \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto\tau \nu\mathbf{c}.(P' \mid R(n,\mathbf{c})+R'(n,\mathbf{c}))$, then
(iii) $\nu\mathbf{c}.(Q \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto\tau Q''$, and
$\nu\mathbf{c}.(P' \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \sim Q''$. Observe that it has to be:
(iv) $Q \mapsto\tau Q'$, and $\nu\mathbf{c}.(Q' \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \equiv Q''$, because:
(a) R(n,c)+R'(n,c) cannot generate a τ-action and simulate.
(b) Q and R(n,c)+R'(n,c) cannot communicate and simulate.

Hence: $\Gamma, \mathbf{c} \supset P' S Q'$.

- $\alpha \equiv d?$, $\Gamma' \equiv x{:}Ch$.

  - $d \notin \mathbf{c}$.

    (i) If $P \mapsto^{d?} x \supset P'$, then

    (ii) $\nu \mathbf{c}.(P \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto^{d?} x \supset \nu \mathbf{c}.(P' \mid R(n,\mathbf{c})+R'(n,\mathbf{c}))$, then

    (iii) $\nu \mathbf{c}.(Q \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto^{d?} x \supset Q''$, and any x
    $x \supset \nu \mathbf{c}.(P' \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \sim Q''$. Observe that it has to be:

    (iv) $Q \mapsto^{d?} x \supset Q'$, and $\nu \mathbf{c}.(Q' \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \equiv Q''$, because
    $R(n,\mathbf{c})+R'(n,\mathbf{c})$ cannot generate a d? action.

Now to check $x \supset P' S Q'$, any x, it is enough to check all the names free in P, Q plus x (see section 5 for a precise explanation). *In particular* we can keep the instance of x distinguished from the fresh names. Hence the thesis.

  - $d \in \mathbf{c}$.

    (i) If $P \mapsto^{d?} x \supset P'$, then we consider $R(n,\mathbf{c}) \equiv c!d'.(R(n+1, \mathbf{c}) + a_n!)$. We have:

    (ii) $\nu \mathbf{c}.(P \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto^{\tau} \nu \mathbf{c}.([d'/x]P' \mid R(n+1, \mathbf{c}) + a_n!)$. Hence:

    (iii) $\nu \mathbf{c}.(Q \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto^{\tau} Q''$, and
    $\nu \mathbf{c}.([d'/x]P' \mid R(n+1, \mathbf{c}) + a_n!) \sim Q''$. Observe that it has to be:

    (iv) $Q \mapsto^{d?} x \supset Q'$, and $\nu \mathbf{c}.([d'/x]Q' \mid R(n+1, \mathbf{c}) + a_n!) \equiv Q''$, because:

    (a) $R(n,\mathbf{c})+R'(n,\mathbf{c})$ cannot generate a τ-action and simulate.

    (b) If the τ-action was generated by Q alone then Q'' could not perform $a_n!$.

    Now observe that we can show $[d'/x]P' S [d'/x]Q'$ for all needed d'.

- $\alpha \equiv c!d$

  - $c \notin \mathbf{c}$, $d \notin \mathbf{c}$, Γ' empty.

    (i) If $P \mapsto^{c!d} P'$, then:

    (ii) $\nu \mathbf{c}.(P \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto^{c!d} \nu \mathbf{c}.(P' \mid R(n,\mathbf{c})+R'(n,\mathbf{c}))$. Hence:

    (iii) $\nu \mathbf{c}.(Q \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto^{c!d} Q''$, and by lemma 3.4.2:
    $\nu \mathbf{c}.(P' \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \sim Q''$. Observe that it has to be:

    (iv) $Q \mapsto^{c!d} Q'$, and $\nu \mathbf{c}.(Q' \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \equiv Q''$, because:
    $R(n,\mathbf{c})+R'(n,\mathbf{c})$ cannot generate a c!d action.

  - $c \notin \mathbf{c}$, $d \in \mathbf{c}$, Γ' empty.

    (i) If $P \mapsto^{c!d} P'$, then, say $\mathbf{c} \equiv d, \mathbf{c}'$

    (ii) $\nu d.\nu \mathbf{c}'.(P \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto^{c!d} d \supset \nu \mathbf{c}'.(P' \mid R(n,\mathbf{c})+R'(n,\mathbf{c}))$. Hence:

    (iii) $\nu d.\nu \mathbf{c}'.(Q \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto^{c!d} d \supset Q''$, and:
    $\nu d.\nu \mathbf{c}'.(P' \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \sim \nu d.Q''$. Observe that it has to be:

    (iv) $Q \mapsto^{c!d} Q'$, and $\nu \mathbf{c}'.(Q' \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \equiv Q''$, because:
    $R(n,\mathbf{c})+R'(n,\mathbf{c})$ cannot generate a c!d action.

- $c \notin \mathbf{c}$, $d \notin \mathbf{c}$, $\Gamma' \equiv$ d:Ch.

(i) If $P \mapsto c!d\ d \supset P'$, then we consider $R(n,\mathbf{c}) \equiv a!c_1 ... a!c_m$. Then:

(ii) $\nu c.(P \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto c!d\ d \supset \nu c.(P' \mid R(n,\mathbf{c})+R'(n,\mathbf{c}))$. Hence:

(iii) $\nu c.(Q \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto c!d\ d \supset Q''$, and
$\nu d.\nu c.(P' \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mid [d/x]R_1) \sim \nu d.(Q'' \mid [d/x]R_1)$, any R_1. Observe:

(iv) $Q \mapsto c!d\ d \supset Q'$, and $\nu c.(Q' \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \equiv Q''$, because:
$R(n,\mathbf{c})+R'(n,\mathbf{c})$ cannot generate a c!d action.

We now take $R_1 \equiv a?x_1 ... a?x_m$. $(R(n+1,\mathbf{x}, d) + an!)$. We have:
$\nu d.\nu c.(P' \mid R(n,\mathbf{c})+R'(n,\mathbf{c}) \mid R_1(d)) \mapsto^{\tau m} \nu d.\nu c.(P' \mid R(n+1,\mathbf{c}, d) + an!)$. Hence:
$\nu d.\nu c.(Q' \mid R(n,\mathbf{c})+R'(n,\mathbf{c}) \mid R_1(d)) \mapsto^{\tau m} \nu d.\nu c.(Q' \mid R(n+1,\mathbf{c}, d) + an!)$, and
$\nu d.\nu c.(P' \mid R(n+1,\mathbf{c}, d) + an!) \sim \nu d.\nu c.(Q' \mid R(n+1,\mathbf{c}, d) + an!)$.

- $c \in \mathbf{c}$, $d \notin \mathbf{c}$, Γ' empty.

(i) If $P \mapsto c!d\ P'$, then we consider $R(n,\mathbf{c}) \equiv c?x.(R(n+1, \mathbf{c}) + an!+x!bn!)$. We have:

(ii) $\nu c.(P \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto \tau\ \nu c.(P' \mid R(n+1, \mathbf{c}) + an!+d!bn!)$. Hence:

(iii) $\nu c.(Q \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto \tau\ Q''$, and
$\nu c.(P' \mid R(n+1, \mathbf{c}) + an!+d!bn!) \sim Q''$. Observe that it has to be:

(iv) $Q \mapsto c!d\ Q'$, and $\nu c.(Q' \mid R(n+1, \mathbf{c}) + an!+d!bn!) \equiv Q''$, because:
(a) $R(n,\mathbf{c})+R'(n,\mathbf{c})$ cannot generate a τ-action and simulate.
(b) If the τ-action was generated by Q alone then Q'' could not perform an!.
(c) If Q performs c!d' then the sequence d!bn! cannot be simulated by Q''.

- $c \in \mathbf{c}$, $d \in \mathbf{c}$, Γ' empty.

(i) If $P \mapsto c!d\ P'$, then we consider $R(n,\mathbf{c}) \equiv c?x.(R(n+1, \mathbf{c}) + an! + Test(n, \mathbf{c}, x))$.

(ii) $\nu c.(P \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto \tau\ \nu c.(P' \mid R(n+1, \mathbf{c}) + an!+Test(n, \mathbf{c}, d))$. Hence:

(iii) $\nu c.(Q \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto \tau\ Q''$, and
$\nu c.(P' \mid R(n+1, \mathbf{c}) + an!+Test(n, \mathbf{c}, d)) \sim Q''$. Observe that it has to be:

(iv) $Q \mapsto c!d\ Q'$, and $\nu c.(Q' \mid R(n+1, \mathbf{c}) + an!+Test(n, \mathbf{c}, d)) \equiv Q''$, because:
(a) $R(n,\mathbf{c})+R'(n,\mathbf{c})$ cannot generate a τ-action and simulate.
(b) If the τ-action was generated by Q alone then Q'' could not perform an!.
(c) If Q performs c!d', $d \neq d'$ then Test(n, c, d) will detect the difference.

- $c \in \mathbf{c}$, $d \notin \mathbf{c}$, $\Gamma' \equiv$ d:Ch.

(i) If $P \mapsto c!d\ d \supset P'$, then we consider
$R(n,\mathbf{c}) \equiv c?x.(R(n+1, \mathbf{c}, x) + an! + x!bn!+Test(n, \mathbf{c}, x))$. Hence:

(ii) $\nu c.(P \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto \tau\ \nu c.\nu d.(P' \mid R(n+1, \mathbf{c}, d) + an!+d!bn!+Test(n, \mathbf{c}, d))$.

(iii) Hence: $\nu c.(Q \mid R(n,\mathbf{c})+R'(n,\mathbf{c})) \mapsto \tau\ Q''$, and
$\nu c.\nu d.(P' \mid R(n+1, \mathbf{c}, d) + an!+d!bn!+Test(n, \mathbf{c}, d)) \sim Q''$. It has to be:

(iv) $Q \mapsto c!d\ d \supset Q'$, and $\nu c.\nu d.(Q' \mid R(n+1, \mathbf{c}) + an!+d!bn!+Test(n, \mathbf{c}, d)) \equiv Q''$, as:
(a) $R(n,\mathbf{c})+R'(n,\mathbf{c})$ cannot generate a τ-action and simulate.
(b) If the τ-action was generated by Q alone then Q'' could not perform an!.
(c) If Q performs c!d', $d \neq d'$ then either d is global and the sequence d!bn! cannot be simulated, or $d \in \mathbf{c}$ and Test will detect the difference. □

Proofs Section 4

4.3.1 Lemma *(From CH_τ-actions to π-actions)*
If $\vdash \Gamma \supset P{:}Pr$ and $\vdash \Gamma \supset P \mapsto\alpha\ \Gamma' \supset P'$ then:

(τ) $\alpha \equiv \tau$ implies there is $\vdash \langle\Gamma\supset P\rangle \mapsto\tau \langle\Gamma\rangle \supset Q'$ and $\underline{\leq\Gamma\geq} \supset \langle P'\rangle \sim Q'$.

(?) $\alpha \equiv c?$, $\Gamma' \equiv \Gamma, x{:}Pr$ implies there is $\vdash \langle\Gamma\supset P\rangle \mapsto c?\ \langle\Gamma\rangle, x{:}Ch \supset Q'$, and $\underline{\leq\Gamma\geq, x{:}Ch} \supset \langle P'\rangle \sim Q'$.

(!) $\alpha \equiv c!P_1$, $\Gamma' \equiv \Gamma, \Gamma_1$ implies there is $\vdash \langle\Gamma\supset P\rangle \mapsto c!d\ \langle\Gamma\rangle, d{:}Ch \supset Q'$ and $\underline{\leq\Gamma\geq, d{:}Ch} \supset \nu\Gamma_1.(\delta(d?.\langle P_1\rangle) \mid \langle P'\rangle) \sim Q'$.

Proof

We proceed by induction on the derivation of the action α; conventionally we denote with $\langle\alpha\rangle$ the corresponding action in the π-calculus.

(!) We derive: $\Gamma \supset (c!P_1.\ P) \mapsto c!P_1\ \Gamma \supset P$, then we can show:
$\langle\Gamma\rangle \supset \nu d.(c!d.\ \langle P\rangle \mid \delta(d?.\langle P_1\rangle)) \mapsto c!d\ \langle\Gamma\rangle, d{:}Ch \supset \langle P\rangle \mid \delta(d?.\langle P_1\rangle)$.

(?) We derive: $\Gamma \supset c?x.\ P \mapsto c?\ \Gamma, x{:}Pr \supset P$, then we have:
$\langle\Gamma\rangle \supset c?x.\ \langle P\rangle \mapsto c?\ \langle\Gamma\rangle, x{:}Ch \supset \langle P\rangle$.

(+.left) We derive: $\Gamma \supset (P_1 + P_2) \mapsto\alpha\ \Gamma' \supset P_1'$, by $\Gamma \supset P_1 \mapsto\alpha\ \Gamma' \supset P_1'$.
Case (i) $\alpha \equiv \tau$ or $\alpha \equiv c?$. Then by ind. hyp. $\langle\Gamma\rangle \supset \langle P_1\rangle \mapsto\langle\alpha\rangle \langle\Gamma'\rangle \supset Q_1'$, and
$\underline{\leq\Gamma'\geq} \supset \langle P_1'\rangle \sim Q_1'$.
Case (ii) $\alpha \equiv c!P''$, $\Gamma' \equiv \Gamma, \Gamma_1$. Then by ind. hyp.
$\vdash \langle\Gamma\rangle \supset \langle P_1\rangle \mapsto c!d\ \langle\Gamma\rangle, d{:}Ch \supset Q_1'$ and
$\underline{\leq\Gamma\geq, d{:}Ch} \supset \nu\Gamma_1.(\delta(d?.\langle P''\rangle) \mid \langle P_1'\rangle) \sim Q_1'$.

($\mid$.left) We derive: $\Gamma \supset (P_1 \mid P_2) \mapsto\alpha\ \Gamma' \supset P_1' \mid P_2$, by $\Gamma \supset P_1 \mapsto\alpha\ \Gamma' \supset P_1'$.
Case (i) $\alpha \equiv \tau$ or $\alpha \equiv c?$. Then by ind. hyp.
$\langle\Gamma\rangle \supset \langle P_1\rangle \mapsto\langle\alpha\rangle \langle\Gamma'\rangle \supset Q_1'$, and $\underline{\leq\Gamma'\geq} \supset \langle P_1'\rangle \sim Q_1'$. That implies:
$\underline{\leq\Gamma'\geq} \supset \langle P_1'\rangle \mid \langle P_2\rangle \sim Q_1' \mid \langle P_2\rangle$.
Case (ii) $\alpha \equiv c!P''$, $\Gamma' \equiv \Gamma, \Gamma_1$. Then by ind. hyp.
$\vdash \langle\Gamma\rangle \supset \langle P_1\rangle \mapsto c!d\ \langle\Gamma\rangle, d{:}Ch \supset Q_1'$ and
$\underline{\leq\Gamma\geq, d{:}Ch} \supset \nu\Gamma_1.(\delta(d?.\langle P''\rangle) \mid \langle P_1'\rangle) \sim Q_1'$. That implies:
$\underline{\leq\Gamma\geq, d{:}Ch} \supset \nu\Gamma_1.(\delta(d?.\langle P''\rangle) \mid \langle P_1'\rangle \mid \langle P_2\rangle) \sim \nu\Gamma_1.(\delta(d?.\langle P''\rangle) \mid \langle P_1'\rangle) \mid \langle P_2\rangle \sim Q_1' \mid \langle P_2\rangle$.

(δ) We derive: $\Gamma \supset \delta P \mapsto\alpha\ \Gamma' \supset P' \mid \delta P$, by $\Gamma \supset P \mid P \mapsto\alpha\ \Gamma' \supset P'$.
Case (i) $\alpha \equiv \tau$ or $\alpha \equiv c?$. Then by ind. hyp.
$\langle\Gamma\rangle \supset \langle P \mid P\rangle \mapsto\langle\alpha\rangle \langle\Gamma'\rangle \supset Q'$, and $\underline{\leq\Gamma'\geq} \supset \langle P'\rangle \sim Q'$.
Case (ii) $\alpha \equiv c!P''$, $\Gamma' \equiv \Gamma, \Gamma_1$. Then by ind. hyp.
$\vdash \langle\Gamma\rangle \supset \langle P \mid P\rangle \mapsto c!d\ \langle\Gamma\rangle, d{:}Ch \supset Q'$ and
$\underline{\leq\Gamma\geq, d{:}Ch} \supset \nu\Gamma_1.(\delta(d?.\langle P''\rangle) \mid \langle P'\rangle) \sim Q'$. That implies:
$\underline{\leq\Gamma\geq, d{:}Ch} \supset \nu\Gamma_1.(\delta(d?.\langle P''\rangle) \mid \langle P'\rangle \mid \delta\langle P\rangle) \sim Q' \mid \delta\langle P\rangle$.

(ν) We derive: $\Gamma \supset \nu c'.P \mapsto\alpha \Gamma,\Gamma' \supset \nu c'.P'$, by $\Gamma,c'\colon Ch \supset P \mapsto\alpha \Gamma,c'\colon Ch,\Gamma' \supset P'$, $c' \notin \alpha$.
Case (i) $\alpha \equiv \tau$ or $\alpha \equiv c?$. Then by ind. hyp.
$\langle\Gamma\rangle,c'\colon Ch \supset \langle P\rangle \mapsto\langle\alpha\rangle \langle\Gamma\rangle,c'\colon Ch,\Gamma' \supset Q'$, and
$\underline{\leq\Gamma\rangle,c'\colon Ch,\Gamma'} \supset \langle P'\rangle \sim Q'$. That implies:
$\underline{\leq\Gamma\rangle,\Gamma'} \supset \nu c'.\langle P'\rangle \sim \nu c'.Q'$.
Case (ii) $\alpha \equiv c!P''$, $\Gamma' \equiv \Gamma, \Gamma_1$. Then by ind. hyp.
$\vdash \langle\Gamma\rangle,c'\colon Ch \supset \langle P\rangle \mapsto c!d \langle\Gamma\rangle,c'\colon Ch, d\colon Ch \supset Q'$ and
$\underline{\leq\Gamma\rangle,c'\colon Ch,d\colon Ch} \supset \nu\Gamma_1.(\delta(d?.\langle P''\rangle) \mid \langle P'\rangle) \sim Q'$. That implies:
$\underline{\leq\Gamma\rangle,\Gamma' ,d\colon Ch} \supset \nu c'.\nu\Gamma_1.(\delta(d?.\langle P''\rangle) \mid \langle P'\rangle) \sim \nu c'.Q'$.

(ν.open) We derive: $\Gamma \supset \nu c'.P \mapsto c!P''$ $\Gamma,c'\colon Ch,\Gamma'' \supset P'$, by
$\Gamma,c'\colon Ch \supset P \mapsto c!P''$ $\Gamma,c'\colon Ch,\Gamma'' \supset P'$, where: $c' \in P''$, $c' \neq c$. Then by ind. hyp.
$\vdash \langle\Gamma\rangle,c'\colon Ch \supset \langle P\rangle \mapsto c!d \langle\Gamma\rangle,c'\colon Ch, d\colon Ch \supset Q'$ and
$\underline{\leq\Gamma\rangle,c'\colon Ch,d\colon Ch} \supset \nu\Gamma''.(\delta(d?.\langle P''\rangle) \mid \langle P'\rangle) \sim Q'$.

($\mid$.com!?) We derive: $\Gamma \supset P_1 \mid P_2 \mapsto\tau \Gamma \supset \nu\Gamma'.(P_1' \mid [P''/x]\tau P_2')$, by
$\Gamma \supset P_1 \mapsto c!P''$ $\Gamma,\Gamma' \supset P_1'$ and $\Gamma \supset P_2 \mapsto c?$ $\Gamma,x\colon Pr \supset P_2'$. Then by ind. hyp.
$\vdash \langle\Gamma\rangle \supset \langle P_1\rangle \mapsto c!d \langle\Gamma\rangle,d\colon Ch \supset Q_1'$,
$\vdash \langle\Gamma\rangle \supset \langle P_2\rangle \mapsto c? \langle\Gamma\rangle,x\colon Ch \supset Q_2'$, $\underline{\leq\Gamma\rangle,d\colon Ch} \supset Q_1' \sim \nu\Gamma'.(\delta(d?.\langle P''\rangle) \mid \langle P_1'\rangle)$,
and $\underline{\leq\Gamma\rangle,x\colon Ch} \supset Q_2' \sim \langle P_2'\rangle$. *By lemma 4.2.3:*
$\langle\Gamma\rangle \supset \nu\Gamma'.(\langle P_1'\rangle \mid \langle[P''/x]\tau P_2'\rangle) \sim \nu\Gamma'.(\langle P_1'\rangle \mid \langle\{P''/x\}P_2'\rangle) \equiv$
$\nu\Gamma'.(\langle P_1'\rangle \mid \nu d.(\delta(d?.\langle P''\rangle) \mid [d/x]\langle P_2'\rangle)) \sim \nu d.(Q_1' \mid [d/x]Q_2')$. □

4.3.2 Lemma *(From π-actions to CH_τ-actions)*
If $\vdash \Gamma \supset P\colon Pr$, and there is $\vdash \Gamma \supset \langle P\rangle \mapsto\alpha \langle\Gamma\rangle,\Gamma' \supset Q'$ then:

(τ) $\alpha \equiv \tau$ implies there is $\vdash \Gamma \supset P \mapsto\tau \Gamma \supset P'$ and $\underline{\leq\Gamma\rangle} \supset \langle P'\rangle \sim Q'$.

(?) $\alpha \equiv c?$, $\Gamma' \equiv \Gamma,x\colon Ch$ implies there is $\vdash \Gamma \supset P \mapsto c?\Gamma,x\colon Pr \supset P'$ and $\underline{\leq\Gamma\rangle,x\colon Ch} \supset \langle P'\rangle \sim Q'$.

(!) $\alpha \equiv c!d$ implies $\Gamma' \equiv d\colon Ch$, there is $\vdash \Gamma \supset P \mapsto c!P''$ $\Gamma,\Gamma_1 \supset P'$ and
$\underline{\leq\Gamma\rangle,d\colon Ch} \supset \nu\Gamma_1.(\delta(d?.\langle P''\rangle \mid \langle P'\rangle) \sim Q'$.

(!') $\alpha \equiv x!$ implies $\Gamma' \equiv \varnothing$, $x\colon Pr \in \Gamma$, there is $\vdash \Gamma \supset [x!/x]P \mapsto x! \Gamma \supset [x!/x]P'$ and
$\underline{\leq\Gamma\rangle} \supset \langle P'\rangle \sim Q'$.

Proof

By induction on the length of the derivation of α. We denote with α' the action corresponding to α in CH_τ.

(!) We are in case (!)'. We will disregard to consider this case in the following clauses. Observe that it can arise everywhere but for (ν.open) and ($\mid$.com!?).

(?) If $\Gamma \supset c?x.\ \langle P\rangle \mapsto c? \langle\Gamma\rangle,x\colon Ch \supset \langle P\rangle$, then
$\Gamma \supset c?x.\ P \mapsto c? \Gamma,x\colon Pr \supset P$, and $\underline{\leq\Gamma\rangle,x\colon Ch} \supset \langle P\rangle \sim \langle P\rangle$.

(+.left) We derive: $\langle\Gamma\rangle \supset (\langle P_1\rangle + \langle P_2\rangle) \mapsto\alpha \langle\Gamma\rangle,\Gamma' \supset Q_1'$,
by $\langle\Gamma\rangle \supset \langle P_1\rangle \mapsto\alpha \langle\Gamma\rangle,\Gamma' \supset Q_1'$.
Case (i) $\alpha \equiv \tau$ or $\alpha \equiv c?$. Then by ind. hyp.:
$\Gamma \supset P_1 \mapsto\alpha' \Gamma,\Gamma'' \supset P_1'$, $\langle\Gamma''\rangle \equiv \Gamma'$, $\langle\Gamma\rangle,\Gamma' \supset \langle P_1\rangle \mapsto\alpha \langle\Gamma'\rangle \supset Q_1'$, and

$\underline{\leq\Gamma>,\Gamma'} \supset Q_1' \sim <P_1'>$.

Case (ii) $\alpha \equiv c!d$, $\Gamma' \equiv d{:}Ch$. Then by ind. hyp.

$\vdash \Gamma \supset P_1 \mapsto^{c!P''} \Gamma,\Gamma_1 \supset P_1'$ and

$\underline{\leq\Gamma>,d{:}Ch} \supset \nu\Gamma_1.(\delta(d?.<P''>) \mid <P_1'>) \sim Q_1'$.

($\mid$.left), (δ), (ν): similar to (+.left)

(ν.open) Remark that in the translation all output actions force the application of exactly one (ν.open). On the other hand the translation of restrictions in the original CHOCS program are never opened. The only possibility of applying (ν.open) is that we derive:

$<\Gamma> \supset \nu d.c!d.(\delta(d?.<P''>) \mid <P>) \mapsto^{c!d} <\Gamma>, d{:}Ch \supset (\delta(d?.<P''>) \mid <P>)$, from

$<\Gamma>,d{:}Ch \supset c!d.(\delta(d?.<P''>) \mid <P>) \mapsto^{c!d} <\Gamma>, d{:}Ch \supset (\delta(d?.<P''>) \mid <P>)$. Then:

$\Gamma \supset c!P''.P \mapsto^{c!P''} \Gamma \supset P$, and

$\underline{\leq\Gamma>, d{:}Ch} \supset <P> \mid \delta(d?.<P''>) \sim <P> \mid \delta(d?.<P''>)$.

($\mid$.com!?) We derive: $<\Gamma> \supset <P_1> \mid <P_2> \mapsto^{\tau} <\Gamma> \supset \nu d.(Q_1' \mid [d/x]Q_2')$, by

$<\Gamma> \supset <P_1> \mapsto^{c!d} <\Gamma>,d{:}Ch \supset Q_1'$, $<\Gamma> \supset <P_2> \mapsto^{c?} <\Gamma>,x{:}Ch \supset Q_2'$. By ind. hyp.

(i) $\Gamma \supset P_1 \mapsto^{c!P''} \Gamma,\Gamma_1 \supset P_1'$,

$\underline{\leq\Gamma>,d{:}Ch} \supset Q_1' \sim \nu\Gamma_1.(\delta(d?.<P''>) \mid <P_1'>)$

(ii) $\Gamma \supset P_2 \mapsto^{c?} \Gamma,x{:}Pr \supset P_2'$,

$\underline{\leq\Gamma>,x{:}Ch} \supset Q_2' \sim <P_2'>$. From (ii):

$\underline{\leq\Gamma>,d{:}Ch} \supset [d/x]Q_2' \sim [d/x]<P_2'>$.

Hence by combining (i), (ii):

$\underline{\leq\Gamma>,d{:}Ch} \supset Q_1' \mid [d/x]Q_2' \sim \nu\Gamma_1.(\delta(d?.<P''>) \mid <P_1'>) \mid [d/x]<P_2'>$. That implies:

$\underline{\leq\Gamma>} \supset \nu d.(Q_1' \mid [d/x]Q_2') \sim \nu\Gamma_1.(<P_1'> \mid \nu d.(\delta(d?.<P''>) \mid [d/x]<P_2'>) \sim$

$\nu\Gamma_1.(<P_1'> \mid <[P''/x]\tau P_2'>)$. □

Proofs section 5

5.2.1 Lemma *(from lts to lts_M transitions)*

Suppose that $\vdash \Delta \supset P$: Pr, and σ is a Δ-substitution. If $\vdash \Delta \supset \sigma P \mapsto^{\alpha} \Delta,\Delta_1 \supset P_1$ then there is $\vdash_M \Delta \supset P \mapsto^{\alpha'} \Delta,\Delta_1' \supset P_1'$, s.t. $\sigma P_1' \equiv P_1$ and

- $\alpha \equiv \tau$: $\alpha' \equiv (\tau, \sigma')$, $\sigma \circ \sigma' = \sigma$.
- $\alpha \equiv d?$, $\Delta_1 \equiv x{:}Ch$: $\alpha' \equiv d'?$, $\sigma(d') = d$, $\Delta_1' \equiv \underline{x{:}Ch} \equiv \underline{\Delta_1}$.
- $\alpha \equiv d!c$: $\alpha' \equiv d'!c'$, $\sigma(d') = d$, $\sigma(c') = c$, $\Delta_1' \equiv \Delta_1$.

Proof

By induction on the derivation of $\vdash \Delta \supset \sigma P \mapsto^{\alpha} \Delta,\Delta_1 \supset P_1$.

(!) If $\vdash \Delta \supset \sigma c'!\sigma d'.\sigma P \mapsto^{\sigma c'!\ \sigma d'} \Delta \supset \sigma P$ then $\vdash_M \Delta \supset c'!d'.P \mapsto^{c'!\ d'} \Delta \supset P$. (?) is similar.

($\mid$.left) Suppose we derive $\sigma\Delta \supset \sigma P \mid \sigma Q \mapsto^{\alpha} \Delta,\Delta_1 \supset P_1 \mid \sigma Q$ from $\Delta \supset \sigma P \mapsto^{\alpha} \Delta,\Delta_1 \supset P_1$. By ind. hyp. $\vdash_M \Delta \supset P \mapsto^{\alpha'} \Delta,\Delta_1' \supset P_1'$, and by ($\mid$.left) $\vdash_M \Delta \supset P \mid Q \mapsto^{\alpha'} \Delta,\Delta_1' \supset P_1' \mid Q$. (+.left) and ($\delta$) are similar.

(ν) As usual we suppose that the bound variables do not interfere with the substitutions. Suppose we derive $\Delta \supset \nu l.\sigma P \mapsto^{\alpha} \Delta,\Delta_1 \supset \nu l.P_1$ from $\Delta,l{:}Ch \supset \sigma P \mapsto^{\alpha}$

Δ,l:Ch,$\Delta_1 \supset P_1$. By ind. hyp. $\vdash_M$ Δ,l:Ch$\supset P \mapsto \alpha'$ Δ,l:Ch,$\Delta_1' \supset P_1'$, and by (v) $\vdash_M$ $\Delta \supset \nu l.P \mapsto \alpha'$ $\Delta,\Delta_1' \supset \nu l.P_1'$.

(v.open) Suppose we derive $\Delta \supset \nu l.\sigma P \mapsto^{\sigma c'!l}$ Δ,l:Ch$\supset P_1$ from Δ,l:Ch$\supset \sigma P \mapsto^{\sigma c'!l}$ Δ,l:Ch$\supset P_1$. By ind. hyp. $\vdash_M$ Δ,l:Ch$\supset P \mapsto^{c'!l}$ Δ,l:Ch$\supset P_1'$, and by (v.open) $\vdash_M$ $\Delta \supset \nu l.P \mapsto^{c'!l}$ Δ,l:Ch$\supset P_1'$.

(|.com!?) Suppose we derive $\Delta \supset \sigma P \mid \sigma Q \mapsto \tau$ $\Delta \supset \nu \Delta_1'.(P_1 \mid [\sigma c'/v]Q_1)$ from $\Delta \supset \sigma P \mapsto^{\sigma d'!\ \sigma c'}$ $\Delta,\Delta_1 \supset P_1$, and $\Delta \supset \sigma Q \mapsto^{\sigma d''?}$ Δ,v:Ch$\supset Q_1$, where $\sigma d' = \sigma d''$. By ind. hyp. $\vdash_M \Delta \supset P \mapsto^{d'!\ c'}$ $\Delta,\Delta_1 \supset P_1'$, and $\vdash_M \Delta \supset Q \mapsto^{d''?}$ Δ,<u>v:Ch</u>$\supset Q_1'$. Since σ is a Δ-sub. we may suppose, for instance, $d' <_\Delta d''$, $d'' \in ?(\Delta)$. Hence by $(|.com!?)_M$ $\vdash_M \Delta \supset P \mid Q \mapsto (\tau,[d'/d''])$ $\Delta \supset \nu \Delta_1 .(P_1' \mid [c'/v]Q_1')$. □

5.2.2 Lemma *(from lts_M to lts transitions)*

Suppose $\vdash \Delta \supset P$: Pr and $\vdash_M \Delta \supset P \mapsto \alpha$ $\Delta,\Delta' \supset P'$. Then

- $\alpha \equiv (\tau, \sigma)$: there is $\vdash \Delta \supset \sigma P \mapsto \tau$ $\Delta \supset \sigma P'$.
- o.w.: there is $\vdash \Delta \supset P \mapsto \alpha$ $\Delta,\Delta' \supset P'$ (with Δ' not underlined).

Proof

By induction on the derivation of $\vdash_M \Delta \supset P \mapsto \alpha$ $\Delta,\Delta' \supset P'$.

$(?)_M$ Suppose $\vdash_M \Delta \supset c?v.P \mapsto c?$ Δ,<u>v:Ch</u>$\supset P$ then $\vdash \Delta \supset c?v.P \mapsto c?$ Δ,v:Ch$\supset P$. (!) is similar.

(|.left) Suppose $\Delta \supset P \mid Q \mapsto \alpha$ $\Delta,\Delta' \supset P' \mid Q$ from $\Delta \supset P \mapsto \alpha$ $\Delta,\Delta' \supset P'$. Then by ind. hyp.: (i) $\alpha \equiv (\tau, \sigma)$: $\vdash \Delta \supset \sigma P \mapsto \tau$ $\Delta \supset \sigma P'$, and by (|.left) $\vdash \Delta \supset \sigma P \mid \sigma Q \mapsto \tau$ $\Delta \supset \sigma P' \mid \sigma Q$. (ii) o.w. $\vdash \Delta \supset P \mapsto \alpha$ $\Delta,\Delta' \supset P'$, and by (|.left) $\vdash \Delta \supset P \mid Q \mapsto \alpha$ $\Delta,\Delta' \supset P' \mid Q$. (+.left) and ($\delta$) are similar.

(v) Suppose $\Delta \supset \nu c.P \mapsto \alpha$ $\Delta,\Delta' \supset \nu c.P'$ from Δ,c:Ch$\supset P \mapsto \alpha$ Δ,c: Ch,$\Delta' \supset$ P'. Then by ind. hyp.: (i) $\alpha \equiv (\tau, \sigma)$: $\vdash \Delta$,c:Ch$\supset \sigma P \mapsto \tau$ Δ,c:Ch$\supset \sigma P'$, and by (v) $\vdash \Delta \supset \nu c.\sigma P \mapsto \tau$ $\Delta \supset \nu c.\sigma P'$. (ii) o.w. $\vdash \Delta$,c:Ch$\supset P \mapsto \alpha$ Δ,c:Ch,$\Delta' \supset P'$, and by (v) $\vdash \Delta \supset \nu c.P \mapsto \alpha$ $\Delta,\Delta' \supset \nu c.P'$.

(v.open) Suppose $\Delta \supset \nu c.P \mapsto d!c$ Δ,c:Ch$\supset P'$ from Δ,c:Ch$\supset P \mapsto d!c$ Δ,c: Ch$\supset$ P'. Then by ind. hyp. $\vdash \Delta$,c:Ch$\supset P \mapsto d!c$ Δ,c: Ch$\supset$ P', and by (v.open) $\vdash \Delta \supset \nu c.P \mapsto d!c$ Δ,c:Ch$\supset P'$.

$(|.com!?)_M$ Suppose $\Delta \supset P \mid Q \mapsto (\tau,\sigma)$ $\Delta \supset \nu \Delta'.(P' \mid [c/v]Q')$ from $\Delta \supset P \mapsto d'!c$ $\Delta,\Delta' \supset P'$, $\Delta \supset Q \mapsto d''?$ Δ,<u>v:Ch</u>$\supset Q'$, and Match(d', d'', Δ) = (true, σ). By ind. hyp. $\vdash \Delta \supset P \mapsto d'!c$ $\Delta,\Delta' \supset P'$, $\vdash \Delta \supset Q \mapsto d''?$ Δ,v:Ch$\supset Q'$. This implies: $\vdash \Delta \supset \sigma P \mapsto \sigma d'!\sigma c$ $\Delta,\Delta' \supset \sigma P'$, $\vdash \Delta \supset \sigma Q \mapsto \sigma d''?$ Δ,v:Ch$\supset \sigma Q'$. Conclude by (|.com!?) $\Delta \supset \sigma P \mid \sigma Q \mapsto \tau$ $\Delta \supset \nu \Delta'.(\sigma P' \mid [\sigma c/v]\sigma Q')$. □

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique

ISSN 0249 - 6399